대한민국특허청
KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : 10-2003-0057036
Application Number

출 원 년 월 일 : 2003년 08월 18일
Date of Application AUG 18, 2003

출 원 인 : 김민경
Applicant(s) KIM, MIN KYOUM

2004 년 04 월 23 일

특 허 청
COMMISSIONER

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

【서지사항】

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0002
【제출일자】	2002.12.26
【발명의 명칭】	키패드에서의 알파벳 입력장치 및 그 방법
【발명의 영문명칭】	APPARATUS AND METHOD FOR INPUTTING ALPHABET CHARACTERS
【출원인】	
【성명】	김민경
【출원인코드】	4-1999-033541-8
【발명자】	
【성명】	김민경
【출원인코드】	4-1999-033541-8
【취지】	특허법 제42조의 규정에 의하여 위와 같이 출원합니다. 출원인 김민경 (인)

【수수료】

【기본출원료】	20 면	29,000 원
【가산출원료】	269 면	269,000 원
【우선권주장료】	0 건	0 원
【심사청구료】	0 항	0 원
【합계】	298,000 원	
【감면사유】	개인 (70%감면)	
【감면후 수수료】	89,400 원	

【요약서】

【요약】

본 발명에서 자모음분리키패드에서 (언어제한) 반복선택방법을 적용하여, 어떤 언어에서는 모호성없이 혹은 대부분의 언어에서 적은 모호성으로 입력하고자 하는 어구를 입력할 수 있음을 보였다.

또한 언어제한 입력방법을 풀입력방법으로 하는 병행입력방법(즉, 언어제한 병행입력방법)을 적용하여 특정 어구의 입력초기에 입력값이 심프코드인지 아니면 풀코드인지를 시스템이 효율적으로 인식하여 처리할 수 있는 효율적인 시스템을 제시하였다.

【대표도】

도 6

【색인어】

키패드; 알파벳입력;

【명세서】

【발명의 명칭】

키패드에서의 알파벳 입력장치 및 그 방법 {APPARATUS AND METHOD FOR INPUTTING ALPHABET CHARACTERS ON KEYPAD }

【도면의 간단한 설명】

도1.

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<2> 본 발명은 키패드에서의 알파벳 입력장치 및 그 방법에 관한 것으로서, 특히 전화기 자판과 같은 적은 수의 키를 지닌 키패드에서의 알파벳 입력장치 및 그 방법에 관한 것이다.

【발명이 이루고자 하는 기술적 과제】

<3> 이동 통신의 발전에 따라, 음성 통화 위주의 휴대용 단말기에 문자 등의 디지털 정보를 송수신하는 기능이 추가되었다. 따라서, 초기에는 전화번호 입력을 목적으로 휴대용 단말기에 구비된 키패드는 문자를 입력하는 수단도 포함하게 되었다. 그런데, 휴대용 단말기의 입력 수단으로 사용되는 키패드의 크기가 점점 작아지므로 키패드에 포함된 버튼의 수는 그 한계를 지니게 된다. 한편, 각 언어의 알파벳은 키패드에 포함된 12개의 키의 수를 크게 상회하고

있다. 따라서, 전화기 키패드를 이용하여 하나의 알파벳을 입력하기 위해서는 키패드 상의 버튼들을 하나 또는 2 이상 조합하여 그 알파벳을 표현하여야 한다.

【발명의 구성】

<4> 출원인의 선출원(출원번호 10-2000-0031879 및 PCT/KR00/00601)에서 제시하였던, 발명을 요약하면 다음과 같다.

<5> 먼저 "부분전체선택방법"에 의하면, 키패드 상의 각 버튼에 키패드 내의 전체 버튼들의 배치에 대응하는 형태로 소정수의 격자들을 구성하여 알파벳들을 격자들에 배치하고, 입력하고자 하는 알파벳(이하 "타겟알파벳"으로 약칭)이 배치된 제1 버튼와 제1 버튼의 격자들에서 그 알파벳의 배치 위치에 대응하여 키패드 상의 제2 버튼를 조합하여 누름으로써, 원하는 알파벳을 입력하게 된다. 예를 들어 도1-1에서 "A = [1]+[2] 로 입력하는 것이다.

<6> 버튼의 격자들 중에서 제1버튼과 제2버튼이 동일하게 되는 기준격자를 포함하여 일부의 격자를 사용하는 것, 일부의 격자를 사용하되, 제1버튼과 제2버튼의 조합의 편리도가 높은 순서의 격자를 우선 사용하는 것이 부분전체선택방법의 핵심이었다. 이렇게 부분전체선택방법의 핵심은 기준격자에 있고, 기준격자의 개념을 활용할 수 있는 키패드를 "기준키패드"라고 이름 붙였다.

<7> 다음으로 기준반복선택방법은 부분전체선택방법을 적용할 수 있도록 구성된

기준키패드에서 기준격자위치의 알파벳으로부터 기준격자에 가까운 순서(부분전체선택방법에서의 버튼조합의 편리도 순서)로 버튼의 누름횟수에 따라 알파벳이 선택되도록 하는 방법이었다. 기준반복선택방법은 기준키패드에서 반복선택방법을 적용하는 것인데, 기준키패드를 반복선택방법 전용화하는 것을 "단순키패드"라고 이름 붙였으며, 편의상 기존에 사용되고 있는 것처럼 단순키패드에서 반복선택방법을 적용하는 것을 단순반복선택방법 이라고 이름 붙였다.

<8> 그리고 "컨트롤처리방법"이 있었으며, 여기에는 "첨자컨트롤처리방법" 그리고 "후속컨트롤처리방법"이 있었다. 첨자컨트롤처리방법은 첨자와 기본알파벳으로 이루어진 변형알파벳 입력시, 이를 첨자컨트롤과 기본알파벳을 조합하여 변형알파벳을 입력할 수 있도록 하는 방법이었다. 후속컨트롤처리방법은 어느 버튼에 배정되는 한 그룹의 알파벳들을 대표알파벳과 후속알파벳의 관계로 설정하고, 후속알파벳 입력시, 대표알파벳과 대표알파벳에 부속된 부속순위를 조합하여 후속알파벳을 입력할 수 있도록 하는 것이었다. 예를 들어 도4-1에서 "ㅋ = ㄱ+[*] " 로 입력할 수 있는 것이다.

<9> 첨자컨트롤처리와 후속알파벳컨트롤처리는 본질적으로 유사하며 후속컨트롤처리가 더 일반적인 것으로 보면된다. 첨자컨트롤처리의 경우도 특정 알파벳 그룹에서 기본알파벳에 변형알파벳이 일정한 순서로 부속되어 있는 것으로 볼 수 있기 때문이다. 단 첨자컨트롤처리의 경우 변형알파벳이 첨자와 기본알파벳으로 분해될 수 있기 때문에, 알파벳 그룹이 그 모양에 있어서 강한 연관을 가지고 있는 경우인 것이고 후속컨트롤은 그 순서 혹은 발음 등에 있어서 강한 연관을 가지고 있는 경우이다.

<10> 컨트롤처리방법을 적용하는 장점은 기본알파벳과 후속알파벳(혹은 변형알파벳)과의 연관관계를 통하여 후속알파벳(혹은 변형알파벳)을 키패드상에 표시하지 않음으로, 키패드의 배치를 단순화 시킬 수 있는 것과, 컨트롤처리를 통하여 모호성없이 알파벳을 입력할 수 있는 것이

었다. 후속알파벳을 생략하여 단순화한 키패드를 "후속키패드"라 하였고, 변형알파벳을 생략하여 단순화한 키패드를 "간략키패드"라 하였고 양자를 합하여 "간략화키패드"라 하였다. 후속알파벳(혹은 변형알파벳)을 모두 표시한 키패드를 간략화키패드에 대비하여 "전체표시키패드"라 하였다.

1> 전체표시키패드에서도 컨트롤처리방법을 통하여 후속알파벳(혹은 변형알파벳)을 입력할 수 있다고 하였고, 반대로 간략화키패드에서도 전체표시키패드의 배치를 기억하고 있는 사용자는 전체표시키패드에서의 입력방법을 이용할 수 있다고 하였다. 이렇게 간략화키패드를 확장하여 전체표시키패드를 구성할 수 있고, 전체표시키패드에서도 사용자의 편리에 따라 컨트롤처리방법을 통하여 후속알파벳을 입력할 수 있는 호환성이 선출원의 특징 중의 하나였다.

12> 컨트롤처리방법을 적용하는 효과 중의 하나는, 모호성을 제거할 수 있는 효과 이외에도, 선출원에서 보여준 바와 같이 대표알파벳과 후속알파벳과의 연관관계를 통하여 후속알파벳을 키패드상에 표시하지 않고 "은닉"함으로써, 키패드를 단순화할 수 있는 것이다. 이를 편의상 "은닉형 컨트롤처리방법"이라고 부른다. 그러나 선출원에서 언급한 바와 같이 후속알파벳(혹은 변형알파벳)을 키패드상에 표시한 전체표시키패드에서도 후속알파벳을 컨트롤처리방법을 통하여 입력할 수도 있다. 이를 편의상 "비은닉형 컨트롤처리방법"이라고 부른다.

13> 본 발명의 특성상 구체적인 예를 들어 설명하는 것이 필수적이기 때문에, 상세한 내용은 실시예를 통하여 설명한다.

14> 먼저 각 언어별로 선출원의 내용을 보완하면 다음과 같다. 이하에서 어느 한 언어에서 언급하는 내용중 타 언어에 적용할 수 있는 내용은 타 언어에서 특별히 언급하지 않아도 이를 타 언어에 적용할 수 있음은 자명하다.

1. 공통 보완 사항

1.1 선출원 및 본 발명에서의 키패드의 활용범위

선출원 및 본 발명에서 제시한 키패드는 이동단말기 혹은 표준키보드의 숫자키패드 혹은 스크린상에 소프트웨어적으로 구성되는 키패드 혹은 도어록 등 전화기자판 형태의 모든 분야에 응용될 수 있음은 자명하다. 또한 표준키보드에 구비된 숫자키패드는 선출원 및 본발명에서 제시한 키패드와 숫자 버튼의 배치가 다르나, 선출원 및 본 발명에서의 키패드 버튼상의 배치를 키보드에 구비된 키패드에도 적용할 수 있음은 자명하다. 예를 들어 선출원 및 본 발명에서 [1] 버튼에 배치된 알파벳을 키보드에 구비된 숫자 키패드의 [1] 버튼에 배치하고, 이하 마찬가지 방법으로 배치하여 알파벳입력, 심플코드의 활용 및 각종 코드의 암기용으로 활용할 수 있다.

1.2 연타지연시간 및 이타(離打)지연시간의 설정

한국어, 힌디어 등과 같이 자음과 모음이 번갈아 등장하는 언어에 있어서, 대표자음과 대표모음의 쌍을 각 버튼에 배정하고, 자음은 1타 모음은 2타로 선택하는 방법을 적용함에 있어서, 일정 지연시간(예를 들어 0.1초) 간격 이내에 입력된 동일버튼 2타를 1차적으로 모음으로 인식하도록 알고리즘을 구현하면, 사용자에 거부감을 줄이면서 효율적으로 구현이 가능하다. 이 지연시간은 같은 버튼을 연속적으로 누르는 경우 보편적으로 일정시간 눌러지는 시간간격을 고려하여 결정되어야 할 것이다. 이러한 지연시간을 편의상 "연타지연시간"이라고 부른다. 더불어 일정 지연시간(예를 들어 1초) 간격 이상으로 입력된 동일버튼 2타를 1차적으

로 2개의 자음으로 인식하도록 구현할 수 있다. 이러한 지연시간을 편의상 "이타(離打)지연시간"이라고 부른다. 이는 동일버튼을 3타 이상 누르는 경우에도 적용될 수 있다.

<0> 예를 들어 도4-1 혹은 도4-2에서 [1] + [1] 이 만약 0.08초의 지연시간을 가지고 입력되었을 경우, 이를 모음 "ㅗ"로 1차적으로 인식하고, 만약 1.1 초 지연시간을 가지고 입력되었을 경우 1차적으로 2개의 자음("ㄱ", "ㄱ")으로 인식하는 것이다. 만약 0.5초의 시간지연을 가지고 입력되었을 경우는, 해당언어의 자음과 모음의 출현구조를 통하여 하나의 모음이 입력된 것인지 혹은 두개의 자음이 입력된 것인지를 판단할 수 있다. 또한 0.08초 혹은 1.1초의 지연시간으로 입력되었을 경우에도, 해당언어의 자음과 모음의 출현구조를 통하여 하나의 모음이 입력된 것인지 혹은 두개의 자음이 입력된 것인지를 판단할 최종적으로 판단할 수 있다.

<1> 기존에는 일정 시간간격(예를 들어 1초) 한가지만을 설정하여, 해당 시간 이내에 동일버튼이 연속하여 입력되면 연속 2타로, 해당 시간 이상 지연되어 동일버튼 2타가 입력되면 각기 1타씩으로 인식하도록 되어 있었다. 차이는 연속 2타로 판단하는 기준 시간지연값(예. 0.1초)과 각기 1타씩으로 판단하는 시간지연값(예. 1초)을 다르게 둔 것이다.

<2> 보통의 경우 선출원의 부분전체선택방법과 반복선택방법중 입력의 편리도 면에서 반복선택방법이 우수하다고 볼 수 있다. 따라서 이는 반복선택방법의 장점(입력규칙의 단순성과 편리성)을 활용하면서, 자음과 모음이 번갈아 출현하는 특정언어의 구조를 이용하여 모호성을 대폭회피할 수 있는 것을 의미하며, 연타지연시간과 이타지연시간을 각각 다르게 설정하고, 사용자로 하여금 지정할 수 있도록 함으로써, 알고리즘의 구현을 단순화할 수 있는 장점이 있다.

<3> 1.3 체인형 컨트롤처리방법

선출원에서 첨자컨트롤처리방법은 본질적으로 후속컨트롤처리방법과 같으며 후속컨트롤처리방법이 더 일반적인 방법이라고 하였다. 즉 선출원의 한국어의 예에서 ㄱ, ㅋ, ㄲ 의 관계를 대표알파벳과 변형알파벳의 관계로 설명하여, ㅋ = ㄱ+{격음}, ㄲ = ㄱ+{경음}으로 설명하였으나, ㄱ(대표알파벳), ㅋ(2nd), ㄲ(3rd)와 같이 대표알파벳과 후속알파벳의 관계로 두어도 결과는 마찬가지이기 때문이다.

도2-1의 일본어의 경우, 후속컨트롤처리방법에 의한 입력예를 다시 보이면다음과 같다. あ(대표알파벳), い(2nd), う(3rd), え(4th), お(5th) 와 같이 대표알파벳과 후속알파벳의 관계를 두고, あ 를 [1] 버튼 1타로 선택(あ=[1])할 수 있으며, 2nd, 3rd, 4th, 5th 컨트롤을 임의의 컨트롤버튼(예. [*])에 두고 버튼의 누름횟수에 따라 반복선택하여 대표알파벳과 후속알파벳의 조합으로 후속알파벳을 입력할 수 있다. 편의상 컨트롤을 선택한 상태를 중괄호로 표기하여 예를 들면, 컨트롤 후입력을 적용시, あ=[1], い = あ+{2nd} = [1]+[*], う = あ+{3rd} = [1]+[*]+[*], え = あ+{4th} = [1]+[*]+[*]+[*], お = あ+{5th} = [1]+[*]+[*]+[*]+[*] 와 같이 되는 것이다.

여기서 다시 あ(대표알파벳), い(2nd), う(3rd), え(4th), お(5th) 와 같이 대표알파벳과 후속알파벳의 관계를 두되, 컨트롤버튼에 후속컨트롤을 배치하고 반복선택방법으로 선택하는 것으로 하지 않고, 단지 "후속" 혹은 "다음" 컨트롤을 특정 컨트롤버튼(예. [*])을 한번 눌러 선택할 수 있도록 하고, 2차후속알파벳(예. い)을 입력시 대표알파벳과 "다음컨트롤"의 조합으로 입력하고(즉 い = あ+{다음} = あ+[*]), 3차후속알파벳 え 입력시 선행알파벳(2차후속알파벳)을 3차후속알파벳에 대한 새로운 대표알파벳(2차대표알파벳)으로 간주하여 2차후속알파벳과 "다음컨트롤"을 조합하여 입력할 수 있다. 즉 う = い+{다음} = [1]+[*] + [*] 이 되는 것이다. 마찬가지로 4차후속알파벳 え 입력시 바로 앞의 선행알파벳(3차후속알파벳) う 를 4

차후속알파벳에 대한 새로운 대표알파벳(3차대표알파벳)으로 간주하여 え 입력시 え = う+{다음} = [1]+[*]+[*] + [*] 와 같이 입력할 수 있다. 5차후속알파벳에 대해서도 마찬가지이다.

<7> 이 결과는 컨트롤버튼에 부속순위 컨트롤(2nd, 3rd, 4th, 5th, ...)을 배치하고 이를 반복선택방법에 의하여 선택하도록 하면서, 대표알파벳과 부속순위컨트롤을 조합하여 입력하는 것이나 같은 결과이다. 여기서 1차대표알파벳 あ 를 모호성 없이 선택할 수 있으면, 2차후속알파벳 い 도 모호성 없이 입력할 수 있는 것을 쉽게 알 수 있으며, 나머지 후속알파벳에 대해서도 같다. 즉 え = い+{다음} = [1]+[*] + [*] 와 같이 되는데, い 를 모호성없이 입력할 수 있었고, い 역시 "다음컨트롤"과 조합되는 형태이므로 결과적으로 모호성 없이 후속알파벳인 え를 입력할 수 있는 것이다.

<28> 이렇게 대표알파벳과 후속알파벳의 연관관계에서, 컨트롤버튼에 부속순위컨트롤을 두지 않고, "다음" 컨트롤 하나만 둔 것으로 하고, 후속알파벳 입력시 선행알파벳을 새로운 대표알파벳으로 간주하여 이 선행알파벳과 "다음" 컨트롤을 조합하여 입력하는 방법을 편의상 "체인형 후속컨트롤처리방법"이라고 부른다. 선출원에서 부속순위컨트롤을 후속컨트롤버튼상에 표시하지 않아도, 사용자는 특정 버튼이 후속컨트롤버튼인 것을 인지하고 있으면 된다고 하였으나, 이러한 체인형 후속컨트롤처리방법의 장점은 "다음 컨트롤"을 컨트롤버튼상에 표시하더라도 그 표시가 간략해 질 수 있는 것이다. 본 발명에서는 선출원에서 설명한 "후속컨트롤처리방법"과 "체인형 후속컨트롤처리방법"의 결과가 같으므로 이를 혼용하여 사용한다.

<29> 1.4 건너뛰기 컨트롤처리방법

30> 첨자컨트롤처리방법에서 ".. + e" 로 이루어진 변형알파벳 ë을 입력함에 있어서 이를 .. 과 e 를 각각 조합하여 입력할 수 있음을 보였다. 혹은 e 를 기본알파벳으로 두고 이와 모양 혹은 순위 등에서 연관성을 가지고 있는 나머지 부속된 알파벳을 후속알파벳으로 두어, 부속순위와 기본알파벳의 조합으로 입력할 수도 있었다(예를 들어 e(기본알파벳), /e(2nd), ^e(3rd), . . .). 또한 이렇게 알파벳(즉 특정 숫자버튼)과 조합하여야만 의미를 가질 수 있는 컨트롤들이 배치되는 컨트롤 버튼은 반복적으로 사용하여도 모호성 없이 변형알파벳 혹은 후속알파벳을 입력할 수 있었다.

31> 로마자의 경우 프랑스어에서 11개(/e, ^e, `e, ë , `a, ^a, ^i, `u, ^u, c, ^o)의 변형알파벳이 사용되며, 변형알파벳에 사용된 윗점의 종류는 5가지(/, ^, `, .., s) 이다. 첨자컨트롤의 선택순서를 /, ^, `, .., s 로 하였을 경우, ^a = a+[*]+[*] 로 입력하여야 한다. 그러나, 프랑스어에서 알파벳 a 에는 첨자 "`"와 "^"의 두가지 밖에 결합할 수 없으므로, 첨자 "/"가 붙을 수 없고, 이를 건너뛰어 결합이 가능한 첨자 "^"이 선택되도록 하면, ^a = a+[*] 로 입력할 수 있다. 이를 편의상 "건너뛰기 컨트롤처리방법"이라 부른다. 즉 이는 a (기본알파벳), `a(2nd), ^a(3rd) 와 같이 후속알파벳의 부속순위를 두고 컨트롤처리방법으로 후속알파벳(혹은 변형알파벳)을 입력하는 것과 같다.

32> 마찬가지로 일본어에서 장음이 존재하는 행은 あ행, や행의 알파벳과 た행의 알파벳 중 つ 뿐이고, 탁음이 존재하는 행은 か행, さ행, た행, は행의 알파벳 뿐이고, 반탁음은 오직 は행의 알파벳에만 존재한다. 결국, 알파벳 つ 에는 장음과 탁음의 2가지 변형알파벳이 존재하고, は행의 알파벳에는 탁음과 반탁음의 2가지 변형알파벳이 존재하는 것을 알 수 있다. 따라서, 2가지 변형알파벳이 존재하는 6개의 알파벳을 제외한 나머지 알파벳의 변형알파벳 입력시 첨자컨트롤버튼을 1번 누름으로써 가능하도록 할 수 있다. 예를 들어, 컨트롤버튼을 [*] 버튼

으로 두고, 컨트롤 후입력을 적용시, ぁ = あ+[*], が = か+[*] 와 같이 할 수 있는 것이다. 2가지 변형알파벳이 존재하는 6개의 알파벳의 경우 변형알파벳 입력시는 변형알파벳의 사용빈도순으로 컨트롤이 선택되도록 할 수 있다. 예를 들어, 예를 들어, 컨트롤버튼을 [*] 버튼으로 두고, 컨트롤 후입력을 적용시, っ = つ+[*], づ = つ+[*]+[*], ぼ = ほ+[*], ぽ = ほ+[*]+[*] 와 같이 할 수 있는 것이다. 즉 이는 컨트롤 선택에 있어서, 기본알파벳과 결합할 수 없는 컨트롤은 그 효력을 발생시키지 않도록 하는 것이다. 예를 들어 が = か+[*] 로 하여도 [*]이 장음컨트롤이 아닌 탁음컨트롤을 선택한 것으로 처리할 수 있는 것은 컨트롤과 결합되는 기본알파벳 か에는 장음이 없으므로 장음컨트롤을 선택한 것으로 하지 않고, 건너뛰어 탁음컨트롤을 선택한 것으로 하기 때문이다.

<33> 컨트롤의 선택을 엄격하게 적용할 경우의 장점은 실제로 사용되지 않는 알파벳의 입력도 가능하도록 할 수 있는 것이다. 예를 들어 프랑스어 알파벳 b 에 첨자 ".." 이 붙은 알파벳은 존재하지 않으나, 이러한 알파벳의 입력도 가능하도록 할 수 있으며, 일본어 알파벳 あ에 탁점이 붙은 알파벳은 존재하지 않으나 그러한 알파벳도 입력이 가능하도록 할 수 있는 것이다.

<34> 컨트롤을 반복선택방법으로 선택하더라도 컨트롤처리를 통하여 입력시, 자판에 표시된 대표알파벳을 모호성 없이 입력할 수 있으면(예를 들어 자판에 표시된 대표알파벳이 하나뿐이거나 다수이어도 부분전체선택방법 등 모호성 없는 방식을 적용할 경우), 나머지 후속알파벳도 모호성 없이 입력될 수 있다. 이는 컨트롤처리를 통하여 모호성이 제거되기 때문인데, 컨트롤은 단독으로 특정 알파벳을 나타내지 못하고, 다른 알파벳과 결합하여야 하기 때문인 점에 기인한다.

1.5 숫자 및 영어 알파벳의 컨트롤처리방법에 의한 입력

선출원에서 제시하였던 바, "기준격자에 가까운 순서"로 자국어 알파벳을 배치하고, 다음으로 숫자 그리고 영어알파벳을 배치할 수 있음을 설명하였고, 기준반복선택방법 적용시 역시 기준격자에 가까운 순서로 이를 자국어 알파벳, 숫자가 선택되도록 할 수 있음을 설명하였다. 마찬가지로 후속컨트롤처리방법을 적용함에 있어서, 특정버튼에 속한 자국어 알파벳 뿐만아니라, 숫자, 영어알파벳(로마자 계열언어가 아닌 경우)도 후속컨트롤처리방법을 통하여 입력할 수 있다.

숫자 혹은 영어알파벳을 자국어 후속알파벳에 이어 후속알파벳으로 둘 수도 있다. 편의상 일본어의 경우를 예로 들면, あ(대표알파벳), い(2nd), う(3rd), え(4th), お(5th), 1(6th), .(7th), q(8th), z(9th) 와 같이 둘 수 있는 것이다. 또한 숫자 혹은 영어알파벳을 컨트롤처리하기 위한 가용한 버튼이 있다면, あ(대표알파벳), い(2nd), う(3rd), え(4th), お(5th) 를 위한 컨트롤버튼을 임의의 버튼(예를 들어 [*] 버튼)으로 두고, 다른 버튼(예를 들어 [#] 버튼)을 숫자 혹은 영어알파벳을 위한 버튼으로 두어 あ(대표알파벳), 1(2nd), .(3rd), q(4th), z(5th) 의 관계를 두어 숫자 혹은 영어 알파벳을 입력할 수 있다. 예를 들어 1 = あ+[#] = [1]+[#], . = あ+[#]+[#] = [1]+[#]+[#], q = あ+[#]+[#]+[#] = [1]+[#]+[#]+[#] 과 같이 된다. 컨트롤버튼이 가용하다면, 숫자입력을 위한 컨트롤버튼 그리고 영어알파벳을 위한 컨트롤버튼을 별도로 둘 수도 있다.

이는 타 언어에 있어서도 적용될 수 있고, 후술하는 각종 기호의 입력에도 적용될 수 있다.

1.6 영어알파벳의 발음에 따른 그룹핑

선출원에서 각 언어의 키패드를 구성함에 있어서, 컨트롤처리방법의 적용 및 암기용으로의 사용을 고려하여, 유사 발음군으로 알파벳을 그룹핑하여 각 숫자버튼에 배정하였다. 영어의 경우 현재 사전순서에 따라 3 혹은 4개씩 알파벳을 그룹핑하여 각 그룹을 각 숫자버튼에 배정한 방법이 널리 사용되고 있으나, 마찬가지로 발음의 유사성을 고려하여 그룹핑하고 각 숫자버튼에 배정하는 것이 가능하다. 다음은 발음의 유사성에 따라 영어의 자음을 9개의 그룹으로 그룹핑한 몇가지 사례이다.

B P / C S X / D T / F V H / G K Q / J Z / L R / M N / W Y

B P V / C S X / D T / F H / G K Q / J Z / L R / M W / N Y

영어의 자음을 8개 그룹으로 그룹핑한 사례를 들면 다음과 같다.

B F P V / C G K Q / S X / D T / J Z / L R / M W H / N Y

B F P V / C G K Q / S X / D T / J Z / L R / M N / W Y H

위의 사례 이외에 많은 변형이 가능하다. 5개의 모음은 2개씩 그룹핑된 그룹에 적절히 넣으면 된다. 이는 추후 설명하는 심플코드를 이용한 단축입력방법 적용시 편리를 줄 수 있다. 비영어권 언어의 경우, 이렇게 그룹핑된 영어 알파벳을 각 버튼에 배정함에 있어서, 자국어

발음군과의 유사성을 고려하여 배정할 수 있다. 예를 들어 한국어의 경우 G, K, Q 그룹을 발음이 유사한 알파벳 "ㄱ"이 배정된 버튼에 배정하는 것이다. 일본어의 경우 G, K, Q 그룹을 발음이 유사한 알파벳 "か" 가 배정된 버튼에 배정하는 것이다. 따라서 영어알파벳의 그룹핑에 있어서도, 각 언어의 자국어 알파벳 그룹핑을 고려하여 그룹핑할 수 있다.

<7> 2. 각 언어별 보완사항

<18> 이하에서 각 언어별로 선출원의 내용을 보완하고 발전된 사항을 기술하면 다음과 같다. 각 언어별 보완사항에서 기술된 내용중에도 어느 한 언어에서 설명한 내용이 타 언어에 적용될 수 있는 경우, 이를 특별히 지적하지 않아도 타 언어에서 적용될 수 있음은 자명하다.

<49> 2.1 영어

<50> 선출원에서 전체표시키패드에서 컨트롤처리방법을 통하여 후속알파벳을 입력할 수 있는 것처럼, 영어의 경우에도 대표알파벳 이외의 알파벳을 자판상에 표시하고도 컨트롤처리를 통하여 입력하는 것이 가능하다. 영어의 경우 후속컨트롤(2nd, 3rd)들은 같은 컨트롤버튼상에 배치될 수도 있고(예를 들어 [#] 버튼), 각각을 다른 버튼에 분리하여 배치할 수도 있다(예를 들어 [*], [#] 버튼). 영어의 경우에 ABC가 하나의 버튼에 배치되어 있을 때, A를 대표알파벳으로 하면 B, C가 각각 후속컨트롤처리를 통하여 입력이 되고, B를 대표알파벳으로 하면, A, C가 각각 후속컨트롤처리를 통하여 입력되는 것이다. 선출원에서 언급한 바와 같이 대표알파벳과 후속알파벳의 부속순위는 사용빈도 등을 고려하여 정해질 수 있다.

1> 예를 들어 A, B, C 그룹에서 A를 대표알파벳으로 하고, 2nd, 3rd 컨트롤을 [*] 버튼에 두고, 컨트롤 후 입력을 적용하면, B = A+{2nd} = [2]+[*], C = A+{3rd} = [2]+[*]+[*] 이 된다. 만약 B 를 대표알파벳으로 하고, 2nd, 3rd 컨트롤을 각각 [*] 버튼과 [#] 버튼에 두고, 컨트롤 후 입력을 적용하면, A = B+{2nd} = [2]+[*], C = B+{3rd} = [2]+[#] 이 된다. 도1-2는 더욱 알파벳의 식별성을 용이하게 하기 위하여 각 그룹의 알파벳 중 중앙의 알파벳을 대표알파벳으로 하여 중앙에 두고 좌우에 각각 후속알파벳을 배치한 사례이다. D = E+{2nd} = [3]+[*] 이 된다.

2> 이렇게 대표알파벳만을 대표알파벳이 속한 버튼 1타로 입력하고 나머지 알파벳을 컨트롤처리방법에 의하여 입력하는 것을 편의상 "대표알파벳제외 컨트롤처리방법(CPMERC : Control Processing Method Except Representative Chanracter)" 라고 부르기로 한다.

3> 또한 P, Q, R, S 4개 알파벳이 [7] 버튼에 배정되고, W, X, Y, Z 4개의 알파벳이 [9] 버튼에 배정되는 경우, 선출원의 한국어의 실시예에서 지적한 바와 같이, 4개의 알파벳 중 한 개의 알파벳을 상하인접조합을 이루는 격자에 배치하여 부분전체선택방법을 적용할 수 있다. 도1-3을 참고한다.

4> 2.2 일본어

5> 선출원에서 일본어의 경우에 있어서, 50음도를 기준으로 알파벳을 그룹핑하고, あ단의 알파벳(あ, か, さ, …)을 대표알파벳으로 하고 나머지 알파벳을 후속알파벳으로 하여 후속컨트롤처리할 수 있음을 선출원에서 밝혔다. 선출원에서 제시한 후속알파벳의 부속순위를 적용함에 있어서 선출원의 방법3과 거의 유사하게 다음의 표와 같이 50음도표를 단순히 적용할 수

도 있다. 이는 부속순위의 단순성으로 사용자에 친근감을 줄 수 있다. ん은 선출원에서와 마찬가지로 임의의 대표알파벳에 부속된 것으로 간주하여 처리할 수 있다. 또한 や행 혹은 わ행의 비어있는 곳에는 50음도의 알파벳(예를 들어, い, う, え)이 부속되어 있는 것으로 혹은 ん이 부속되어 있는 것으로 간주할 수도 있다.

6>

선출원의 방법3					50음도 단순활용 방법				
기준격자	2nd	3rd	4th	5th	기준격자	2nd	3rd	4th	5th
あ	い	う	え	お	あ	い	う	え	お
か	き	く	け	こ	か	き	く	け	こ
さ	し	す	せ	そ	さ	し	す	せ	そ
た	ち	つ	て	と	た	ち	つ	て	と
な	に	ぬ	ね	の	な	に	ぬ	ね	の
は	ひ	ふ	へ	ほ	は	ひ	ふ	へ	ほ
ま	み	む	め	も	ま	み	む	め	も
や	ゆ	よ			や		ゆ		よ
ら	り	る	れ	ろ	ら	り	る	れ	ろ
わ	を	ん			わ				を

57> 선출원에서 일본어의 경우에 편의상 가장 대표성을 가진다고 볼 수 있는 1번째 단의 알파벳을 대표알파벳으로 하였는데, 이 역시 임의의 단의 알파벳을 대표알파벳으로 할 수도 있고, 각 그룹에서 임의의 알파벳을 대표알파벳으로 할 수도 있다. 또한 각 행의 알파벳을 그룹으로하여 각 버튼에 배정함에 있어서도, 선출원에서 언급한 바와 같이 키패드상 버튼의 행을 기준([1], [2], [3], [4], . . .)으로 배정할 수도 있고, 열을 기준([3], [6], [9], [2],

[5], . . .)으로 배정할 수도 있다. 마찬가지로 행을 기준으로 혹은 열을 기준으로 배정하지 않고, 임의로 배정할 수도 있다.

8> 또한 선출원의 일본어의 예에서 "ん"을 [0]버튼의 기준격자위치에 두고, [*]버튼과 [#] 버튼에 각각 2nd, 3rd 그리고 4th, 5th 컨트롤을 둔 경우, [0] 버튼을 장음/탁음/반탁음 입력을 위한 컨트롤버튼으로 활용하기 위하여, "ん"을 [0]버튼이 아닌 임의의 숫자버튼에 배정된 것으로 가정하였던 바 있다. 그러나 이 경우는 [0]버튼의 기준격자 위치에 "ん"을 두고, 기준격자에 가까운 순서로 장음/탁음/반탁음 컨트롤을 추가적으로 배치하고 컨트롤 선택에 반복선택방법을 적용하여도 된다. 왜냐하면 "ん"은 한개의 단어에서 연속하여 등장하는 경우가 없기 때문이다. 도2-1를 참고한다. 이렇게 연속하여 등장하지 않는 알파벳을 해당 알파벳이 배치된 버튼 1타에 선택하도록 하고, 2타, 3타, … 에 다른 컨트롤들을 선택할 수 있도록 하는 것은 다른 모든 언어의 경우에도 적용될 수 있다. 후술하는 한국어의 모음요소를 활용한 방법에서도 이러한 성질을 활용하고 있다.

59> 2.3 아랍어

60> 아랍어에는 28개의 자음이 존재한다. 선출원에서 숫자의 의미를 가지는 아랍어 자음을 다음과 같이 그룹핑하고 키패드 버튼상에 배정하고, 각 알파벳을 가장 작은 숫자의 의미를 가지는 알파벳을 각 그룹의 대표알파벳으로 하여 기준격자의 위치에 배치하고 나머지 알파벳을 작은 숫자의 의미를 가지는 알파벳으로부터 기준격자에 가까운 순서(선출원에서 설명)로 키패드상에 배치하는 방안을 제시한 바 있다. 또한 선출원에서 일본어, 로마자계열 언어, 한국어,

인도어, 아랍어 등의 경우에 컨트롤 처리에 의하여 알파벳을 입력하는 방법을 제시한 바 있다. 아랍어의 경우에 있어서도, 별로 사용되지 않는 모음을 첨자로 간주하여 첨자컨트롤처리하는 방법을 제시한 바 있다.

> 여기서는 아랍어의 자음을 컨트롤처리(후속컨트롤처리)하는 방법을 제시하고자 한다. 다음은 선출원에서 아랍어의 자음을 그룹핑하고 각 버튼에 배정한 예이다.

>

버튼	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[0]	
의미	1	2	3	4	5	6	7	8	9	1000	
기준 격자	ا	ب	ج	د	ه	و	ز	ح	ط	غ	1 단위 알파벳
의미	10	20	30	40	50	60	70	80	90		
2nd	ى	ك	ل	م	ن	س	ع	ف	ص		10 단위 알파벳
의미	100	200	300	400	500	600	700	800	900		
3rd	ق	ر	ش	ت	ث	خ	ذ	ض	ظ		100 단위 알파벳
의미	1000										
4th	غ										1000 단위알파벳
	1 그룹 알파벳	2 그룹 알파벳	3 그룹 알파벳	4 그룹 알파벳	5 그룹 알파벳	6 그룹 알파벳	7 그룹 알파벳	8 그룹 알파벳	9 그룹 알파벳	10 그룹 알파벳	

63> [1] 버튼에 배정된 1, 10, 100의 의미를 가지는 알파벳 중 1의 의미를 가지는 알파벳을 대표알파벳으로 하여 키패드상 표시하고 나머지 10, 100의 의미를 가지는 알파벳을 후속컨트롤처리할 수 있다. 컨트롤버튼은 선출원에서와 같이 임의의 버튼을 사용할 수 있다. 후속컨트롤버튼을 [*] 버튼을 사용하고, 2nd, 3rd 컨트롤을 배치할 경우, 자음 입력에 있어서 평균 입력타수는 2타가 된다. 버튼당 하나의 알파벳만이 배치되므로 버튼에 표시된 알파벳을 선택함

에 있어서는 부분전체선택방법으로 선택할 필요는 없다. 알파벳의 배치는 개념적인 것이므로 다음에 설명하는 바와 같이 키패드상에 알파벳을 배치하지 않을 수도 있다.

4> 다시 선출원의 일본어의 경우처럼 후속컨트롤을 다른 버튼으로 분리할 수 있다. 예를 들어 [#] 버튼에 3rd 컨트롤 분산하여 배정한다면, 평균입력타수 약 1.7타((1+2+2)/3)가 되며 28개 자음을 선출원에서 제시한 바와 같이 모호성 없이 입력할 수 있게 된다.

5> 본 출원에서는 각 단위의 알파벳의 사용빈도와 상관없이 숫자의 의미가 적은 알파벳을 각 그룹의 대표알파벳으로 하고 나머지 단위의 알파벳에 있어서도, 적은 숫자의 의미를 가진 알파벳을 먼저 선택할 수 있도록 하는 예를 보인다. 컨트롤은 선입력 혹은 후입력될 수 있다.

66> 후속컨트롤을 모두 [*] 버튼에 배치하고, 1단위의 알파벳을 대표알파벳으로 하고, 10단위알파벳 선택시 컨트롤 버튼 선입력을 적용하면, ㅋ = [*] + ㅜ 가 되고, ㅇ = [*] + [*] + ㅜ가 된다. 나머지 알파벳에 대해서도 동일하다. 이 경우 [#]버튼을 모음컨트롤처리를 위해 사용할 수도 있다.

67> 1000의 의미를 가지는 알파벳 하나만을 [0] 버튼에 배치하면, 입력시 ㆁ = [0] 이 된다. 1000의 의미를 가지는 알파벳을 1그룹알파벳으로 처리할 수도 있다. 이 경우 ㆁ = [*]+[*]+[*]+ㅣ 가 될 것이다. 혹은 1000의 의미를 가지는 알파벳이 70의 의미를 가지는 알파벳(ㅎ)에 윗점이 붙은 형태이므로, 윗점컨트롤을 각종 모음컨트롤에 앞서 선택되도록 하여 컨트롤처리하는 것도 가능하다. 즉 윗점컨트롤 선입력시 ㆁ = [윗점컨트롤] + ㅎ 로 입력할 수 있는 것이다. 윗점이 붙어 이루어진 다른 알파벳에도 이러한 윗점컨트롤처리를 다른 입력

방법과 병행하여 허용할 수 있다. 이 경우 [0] 버튼을 다른 용도의 컨트롤버튼(예를 들어 모음컨트롤버튼)으로 활용하는 것이 용이하게 된다.

8> 만약 컨트롤을 분리하여 3rd 컨트롤(여기서는 100단위컨트롤)을 [#] 버튼으로 하면 200의 의미를 가지는 알파벳 ᄀᆞ = [#] + ᄂᆞ 가 된다. 이 경우도 선출원의 일본어의 경우와 같이 입력시, ᄀᆞ = [*] + [*] + ᄂᆞ 을 병행하여 허용할 수 있다. 1000의 의미를 가지는 알파벳을 1그룹알파벳에 속하게 할 경우, 경우 ᄀᆞᆞᄂᆞ = [*]+[*]+[*]+ ㅣ 혹은 ᄀᆞᆞᄂᆞ = [#]+[#]+ ㅣ 가 된다. 즉 [*] 버튼에 2nd, 3rd, 4th 컨트롤이 배치되어 있는 것으로 그리고 [#] 버튼에 3rd, 4th 컨트롤이 배치되어 있는 것으로 생각하면 된다. 아랍인들이 우에서 좌로 가로쓰기를 하는 점을 고려하여, 즉 [#] 버튼에 2nd, 3rd, 4th 컨트롤을 배정하고, [*] 버튼에 3rd, 4th 컨트롤을 배치하는 것도 가능하다.

69> 이는 아랍어에 있어서, 주로 사용되는 자음을 평균 약 1.7타에 모호성 없이 입력할 수 있는 의미가 있고, 또한 아랍인들은 자국어 알파벳이 가지고 있는 숫자의 의미를 알고 있으므로, 굳이 키패드상에 알파벳(예에서, 1단위알파벳 즉 대표알파벳)을 표시하지 않고도, 알파벳을 입력할 수 있는 매우 큰 장점이 있다. 다시 말하면, 숫자만이 표시된 키패드를 통하여 본 출원에서의 방법을 적용시, 대표알파벳이 어느 단위의 알파벳 인지, 컨트롤버튼이 어느 단위의 알파벳을 위한 컨트롤버튼인지, 후속컨트롤이 하나의 컨트롤버튼을 사용한다면, 각 후속컨트롤의 선택순서는 어떤지, 컨트롤버튼을 선입력하는지 등의 규칙만 숙지하면, 실제 생활에서 필요한 대부분의 알파벳(아랍어 자음)을 입력할 수 있는 것이다.

<70> 선출원에서 지적한 바와 같이 모음을 역시 컨트롤처리할 수 있는데, [#]과 [*]을 모두 자음컨트롤처리를 위해 사용하는 경우, 선출원의 일본어에서와 같이 1000의 의미를 가지는 알파벳을 [0] 버튼에 배치하지 않고, [1] 버튼에 배치하여 컨트롤 처리하여, [0]버튼을 모음처리

용 컨트롤 버튼으로 지정하고 반복선택방법에 의해 첨자형태의 모음(모음컨트롤, 첨자컨트롤)을 선택할 수 있다. 방법은 마찬가지로 사용빈도순으로 버튼의 누름횟수에 따라 모음컨트롤(첨자형태의 모음)이 선택되는 것이다. 자음용 후속컨트롤버튼으로 [*] 하나만 사용된다면, 모음용 컨트롤버튼으로 [#]을 사용할 수도 있고, [#]과 [0]을 모두 사용할 수도 있다.

1> 도3-1, 도3-2의 예시에서는 1 ~ 9의 의미를 가지는 알파벳(1단위 알파벳)을 각 그룹의 대표알파벳으로 하였으나, 각 그룹에서 가장 사용빈도가 많은 알파벳을 대표알파벳으로 할 수도 있다. 그리고 혼동을 줄이기 위하여 1단위알파벳(1 ~ 9의미를 가지는 알파벳), 10단위알파벳(10 ~ 90의미를 가지는 알파벳), 100단위알파벳(100 ~ 900의미를 가지는 알파벳) 중 임의의 단위의 알파벳을 대표알파벳으로 할 수도 있다. 도3-3은 10단위알파벳을 대표알파벳으로 하여 각 그룹의 중앙에 두고 나머지 좌우에 배정된 알파벳(100단위알파벳, 1단위알파벳)을 좌우측에 배정된 컨트롤버튼을 이용하여 컨트롤처리방법에 의하여 입력하는 사례이다.

72> 마찬가지로 후속컨트롤에 의하여 선택되는 알파벳도 사용빈도에 따른 각 단위의 알파벳으로 할 수 있다. 예를 들어 100단위알파벳이 가장 사용빈도가 많아, 이를 대표알파벳으로 하고, 그 다음으로 1단위알파벳이 사용빈도가 많다면, 2nd 컨트롤(즉 여기서는 1단위컨트롤)과 대표알파벳을 조합하여 입력할 수 있도록 하고, 10단위알파벳을 3rd 컨트롤(즉, 10단위컨트롤)과 대표알파벳을 조합하여 입력할 수 있도록 하는 것이다.

73> 선출원에서 전체표시키패드에서도 후속알파벳을 컨트롤처리방법을 통하여 입력하는 것이 가능하다고 하였으므로, 마찬가지로 선출원의 키패드에서도 자음을 후속컨트롤처리방법에 의하여 입력하고, 모음을 첨자컨트롤처리방법에 의하여 입력하는 것이 가능하다. 3*4 키패드 내에서, 모음용 첨자컨트롤버튼을 [0] 버튼으로 하면, 자음 입력을 위한 컨트롤버튼을 [*] 버튼과 [#] 버튼에 분산하는 것이 가능하다.

<4> 2.4 한국어

<5> 2.4.1 각종 컨트롤의 적용

<6> 도4-1 ~ 도4-3은 한국어의 기본자음과 기본모음을 쌍으로 각 버튼에 배정하고, 키패드상 표시된 기본자음과 기본모음을 반복선택방법에 의하여 입력하는 방법이다. 도4-1 에서 격음, 경음 그리고 확장모음을 컨트롤처리방법에 의하여 입력하도록 하였고, 도4-2에서는 격음과 경음만을 컨트롤처리방법에 의하여 입력하도록 하였다. 도4-3은 격음, 경음, 기본모음, 그리고 확장모음을 컨트롤처리방법에 의하여 입력하도록 한 사례이다.

<7> 2.4.2 프로그램 구현

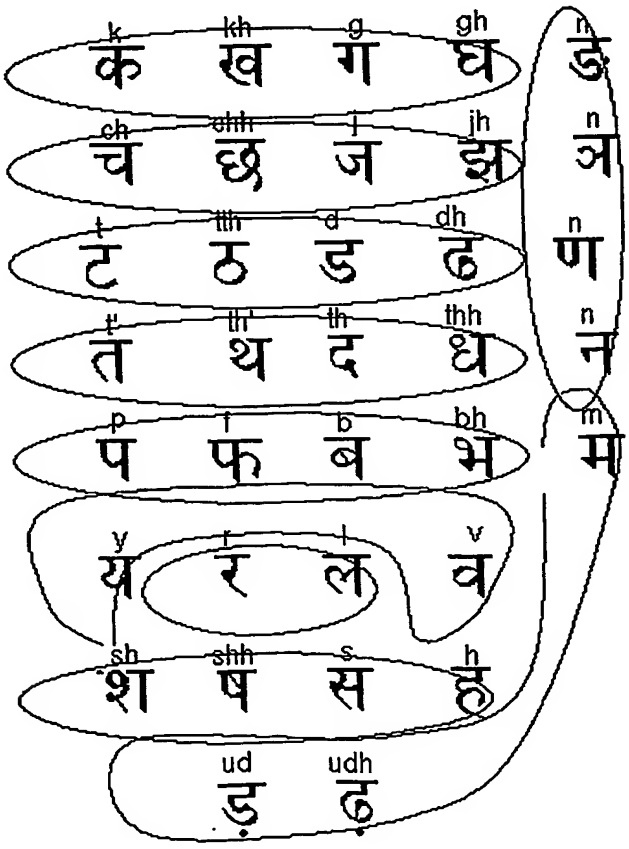
<8> 도4-4는 단지 구현을 위한 흐름도의 일례일 뿐이고 좀 더 효율적인 프로그래밍이 가능하다. 예를 들어 도4-4에서 받침의 자음부분이 왔을 경우 이중받침을 이룰 수 있는 자음인가를 앞서 체크함으로써, 약간 더 효율적인 구현이 가능하다.

<9> 선출원에서 제시한 한국어의 예는 유사한 특징(자음과 모음이 번갈아 출현하는 구조)을 가지는 타 언어에 있어서도 응용될 수 있다. 타 언어의 경우에는 타 언어의 자모출현 특성을 반영하면 된다.

<80> 예를 들어 기본자음과 기본모음의 쌍을 이용하여 힌디어 입력시스템을 구성하고자 하면, 힌디어의 특성을 반영하여 한국어의 경우와 유사하게 구현할 수 있으며, 힌디어의 경우 한국어의 경우보다 자음과 모음의 출현규칙이 간단하므로 더 쉽게 구현할 수 있다.

2.5 힌디어

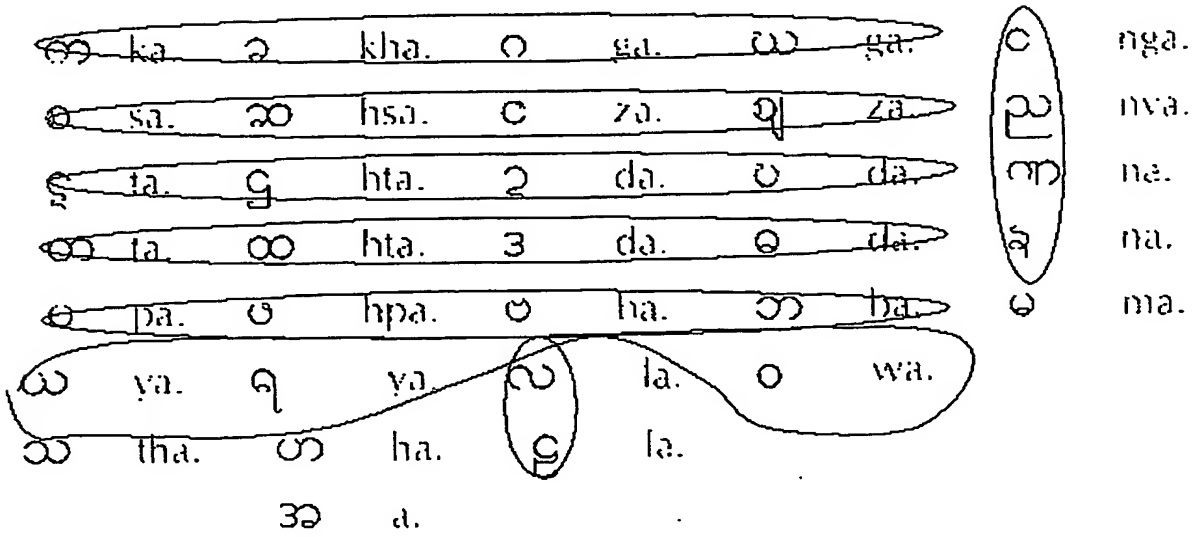
출원인의 선출원에서 힌디어 자음을 9개의 그룹으로 그룹핑하여 [1] ~ [9] 버튼에 배정함으로써, [0] 버튼에 배정된 모음은 [0]버튼 1타로 선택하게 되는데, 일관성을 위하여 자음을 10개의 그룹으로 그룹핑하고 항상 대표자음은 1타로 선택되고 10개의 모음은 2타로 선택되도록 할 수 있다. 키패드상에 배치되지 않고 자주 쓰이지 않는 모음 ___(ri)는 컨트롤처리방법에 의하여 처리할 수 있다. 자음을 10개의 그룹으로 그룹핑 하는데 있어서는 선출원에서 제시한 바와 같이 발음의 유사성을 고려하여 그룹핑하는 방법이 가능하다. 10개의 그룹으로 그룹핑하는 일례를 들면 다음과 같다.



그리고 힌디어의 첫번째 모음 ___(a) 는 자음과 자음 사이에서 생략될 수 있다고 한다. 즉 자음이 연속하여 나오게 되는 것이다. 이러한 경우 "자음 + ___(a) + 자음" 으로 입력하더라도 "자음 + 자음" 으로 표시되도록 하는 것은 매우 용이한 일이다. 물론 직접 "자음 + 자음"을 입력하는 것도 가능하나 이 경우 동일한 대표자음이 연속하여 선택되어 두개의 자음이 하나의 모음으로 인식될 수 있는 가능성이 많아진다. 따라서 양자(모음 ___(a)를 자동 생략 혹은 사용자가 생략 입력)를 모두 허용할 수 있다.

2.6 미얀마어

미얀마어에는 다음의 33개 자음이 존재한다. 이를 힌디어의 예와 같이 9개 혹은 10개의 그룹으로 그룹핑하고 대표알파벳을 정하고, 나머지 자음을 후속컨트롤처리방법에 의하여 입력할 수 있다.



3. 다차원 교차 컨트롤처리방법

9> 도2-2는 위의 표에 의하여 "あ"단의 알파벳을 대표알파벳으로 하여 각 버튼에 사전순서로 배정한 사례이다. 선출원의 사례에서 2nd, 3rd 컨트롤이 배정된 컨트롤버튼에 4nd, 5rd 컨트롤이 추가로 배정될 수도 있었으나, 여기서는 입력타수를 최소화하고, 컨트롤버튼을 최대한 활용하기 위하여 2nd, 3rd 컨트롤만을 배정한 사례를 보인다.

10> 도2-2에서 후속알파벳을 입력하는 방법은 선출원에서 제시한 바와 같다. 예를 들어, 컨트롤 후입력 적용시 い = あ+[*] 와 같이 된다. 다음으로 각 알파벳의 변형알파벳인 장음, 탁음, 반탁음을 입력하는 방법을 보인다. 도2-2에서 컨트롤버튼으로 가용한 버튼([*] 버튼, [#] 버튼)이 모두 후속컨트롤버튼으로 사용되고 있는 상태이다. 따라서 변형알파벳을 위한 컨트롤버튼이 없는 상태이다. 그러나, 타겟알파벳의 기본알파벳이 후속알파벳인 경우 그 후속알파벳 입력후 후속컨트롤버튼으로 사용되지 않은 후속컨트롤버튼을 변형알파벳컨트롤버튼(이를 편의상 "반대편 컨트롤버튼"이라고 부름)으로 활용하여 입력할 수 있다.

91> 예를 들어, い = い + "반대편 컨트롤버튼" = あ+[*] + [#], ご = こ + "반대편 컨트롤버튼" = か+[#]+[#] + [*] 과 같이 된다. 이는 い를 기본알파벳 い 의 변형알파벳으로 간주하여 컨트롤처리방법을 적용함에 있어서, い를 입력하기 위하여 후속컨트롤버튼으로 사용되지 않은 반대편 컨트롤버튼을 변형알파벳컨트롤버튼으로 활용하는 것으로 볼 수 있다. 예에서와 같이 여기서도 선출원의 건너뛰기 컨트롤처리방법을 적용할 수 있다. 이를 체인형 컨트롤처리방법의 관점에서 보면, 반대편 컨트롤버튼을 1번 누름으로써, 기 입력된 알파벳(예에서 こ)의 변형알파벳(예에서 ご)을 입력하기 위한 "다음 컨트롤"을 선택하는 것으로 볼 수 있다.

2> は행의 알파벳에는 변형알파벳이 탁음과 반탁음의 2가지가 있으므로 장음의 경우는 반대편 컨트롤버튼을 1번 누르고 반탁음의 경우는 반대편 컨트롤을 2번 누르도록 할 수 있다. 즉 ぶ = ふ + "반대편 컨트롤버튼" = は+[*]+[*] + [#], ぷ = ふ + "반대편 컨트롤버튼 2번누름" = は+[*]+[*] + [#]+[#] 과 같이 된다.

3> 다음으로 타겟알파벳의 기본알파벳이 대표알파벳인 경우는 위의 방법을 그대로 적용할 수 없다. 이 경우 후속컨트롤버튼 중 임의의 버튼에 추가로 대표알파벳의 변형알파벳을 후속알파벳으로 두어 타겟알파벳을 입력하도록 할 수 있다.

4> 예를 들어, [*] 버튼을 사용하여 대표알파벳의 변형알파벳(장음, 탁음, 반탁음)을 입력하는 경우 あ = あ+[*]+[*]+[*] 와 같이 된다. 즉 대표알파벳의 변형알파벳(장음, 탁음, 반탁음)을 [*] 버튼을 이용하여 입력할 수 있는 4th 후속알파벳으로 간주하는 것이다. 이상의 내용을 정리하면 다음과 같이 된다.

95>

	[*]버튼 활용		[#]버튼 활용		[*]버튼 혹은 [#]버튼 중 임의의 버튼 활용(예. [*] 버튼)							
대표 알파벳	2번	3번	4번	5번	4번	5번	6번	7번		...	...	
あ	い	う	え	お	ぁ	1	영1 .	영2 Q	영3 Z	기호1	...	
	ぃ	ぅ	ぇ	ぉ								반대편 컨트롤버튼 활용
か	き	く	け	こ	が	2	영1 A	영2 B	영3 C	기호1	...	
	ぎ	ぐ	げ	ご								반대편 컨트롤버튼 활용
...												
は	ひ	ふ	へ	ほ	ば	ぱ	6	영1 M	영2 N	영3 O	...	
	び	ぶ	べ	ぼ								반대편 컨트롤버튼 활용
	ぴ	ぷ	ぺ	ぽ								반대편 컨트롤버튼 활용
...												
や	ゆ		よ		ゃ							
	ゅ		ょ									반대편 컨트롤버튼 활용
...												

96> 이렇게 반대편 컨트롤버튼을 활용하여 컨트롤버튼의 용도를 확장하고, 더 많은 알파벳 혹은 기타의 입력에 활용하는 것을 편의상 "교차 컨트롤처리방법" 혹은 "지그재그 컨트롤처리방법" 혹은 "다차원 교차 컨트롤처리방법"이라고 부른다. 이는 3개 이상의 버튼을 컨트롤버튼으로 사용하는 경우에도 적용될 수 있다.

97> 위의 표에서 첫번째 컬럼의 칸을 제외한 나머지 컬럼의 칸 역시 다른 알파벳 혹은 각종의 기호를 입력하기 위한 용도로 활용할 수 있음은 자명하다. 위의 예에서는 반대편 컨트롤버튼을 한번 적용한 것이다. 즉 ぷ 입력시 ぷ = ふ + "반대편 컨트롤버튼 2번누름" = は +[*]+[*] + [#]+[#] 와 같이 반대편 컨트롤버튼을 반복적으로 사용하여 후속컨트롤버튼으로 사용된 [*] 버튼에 대하여 반대편 컨트롤버튼([#] 버튼)을 한번만 적용한 것이다. 이를 편의상 "2차원 교차컨트롤처리" 혹은 "2단계 교차컨트롤처리" 라고 한다.

8> 즉 하나의 알파벳 입력을 위하여 하나의 컨트롤버튼만을 사용하는 경우 이를 "1차원 컨트롤처리"라고 할 수 있다. 이는 컨트롤버튼을 "1차원적"으로 사용한다고 볼 수 있는 것이다. 그런데 2차원 (교차)컨트롤처리방법에서는 하나의 알파벳 입력을 위하여 다른 컨트롤버튼이 2개 사용된다. 즉 ご 입력시 ご 를 こ의 변형알파벳으로 간주하여, ご = こ + "반대편 컨트롤버튼" = か+[#]+[#] + [*] 으로 입력하였는데, 여기서 반대편컨트롤버튼은 이미 대표알파벳 か의 1st, 2nd 후속알파벳인 き, く를 입력하기 위한 컨트롤 버튼으로 정의되어 있었음에도, 마치 반대편 컨트롤버튼([*] 버튼)을 こ의 변형알파벳 ご 를 입력하기 위한 변형알파벳컨트롤버튼과 같이 활용할 수 있는 것이다. 이는 마치 하나의 알파벳 입력을 위하여 하나의 컨트롤버튼을 1차원적으로 사용하는 것에 비하여 2개의 컨트롤버튼을 "2차원적"으로 사용하는 것으로 볼 수 있다.

39> ぷ = ふ + "반대편 컨트롤버튼1번 누름" + "앞의 컨트롤버튼에 대하여 반대편 컨트롤버튼 1번 누름" = は+[*]+[*] + [#]+[*] 과 같이 입력할 수도 있다. 즉 ぷ 의 입력에 최종적으로 사용된 컨트롤버튼(예에서 [#] 버튼)을 반복적으로 사용하여 변형알파벳을 입력하는 것이 가용하지 않을 경우 마찬가지 방법으로 반대편 컨트롤버튼에 대한 반대편 컨트롤버튼을 다시 이용하여 다른 변형알파벳 혹은 후속알파벳을 입력할 수 있는 것이다. 이를 편의상 "3차원 교차컨트롤처리" 혹은 "3단계 교차컨트롤처리"라고 부른다. 이는 교차 컨트롤처리방법을 이용하여 키패드내에서 입력할 수 있는 알파벳 혹은 기타 알파벳 입력과 관련된 것들의 가능한 입력갯수를 무한히 확장할 수 있음을 의미한다. 더구나 선 입력된 알파벳과 관련성을 가진 후속알파벳 혹은 변형알파벳을 자연스럽게 컨트롤처리방법에 의하여 입력할 수 있는 큰 의미가 있다.

100> 이상의 내용을 2가지의 변형알파벳(탁음, 반탁음)이 존재하는 は 행의 예를 들어 그래프의 형태로 설명하면 도2-3과 같다. 도2-3에서 흐린 색으로 동그라미에 ". . ."으로 표시된 것

은, 위의 표에서 빈 공간에 다른 알파벳 혹은 기호를 입력하는데 이용할 수 있는 것과 마찬가지로, 만약 필요시 추가로 입력하고자 하는 것(알파벳 혹은 기호)를 다차원 컨트롤처리방법에 의하여 입력하게 할 수 있음을 의미한다. 도2-3에서 [*] 버튼과 [#] 버튼은 각기 직각 방향으로 사용되고 있다.

1> 선출원에서 [*] 버튼에 50음도의 알파벳 중 대표알파벳이 아닌 2nd, 3rd, 4th, 5th 컨트롤을 배정하고, [*] 버튼을 장음, 탁음, 반탁음 등의 변형알파벳을 입력하기 위한 컨트롤버튼으로 활용하는 방법과 비교해 보면 선출원의 방법에서도 2차원 컨트롤처리가 적용되고 있음을 알 수 있다. 즉 い = い+{변형} = い+[*]+[#] 으로 입력되는데 2개의 컨트롤버튼이 사용되기 때문이다. 차이는 선출원에서는 컨트롤버튼의 용도가 후속컨트롤(2nd, 3rd, 4th, 5th)버튼으로만 그리고 변형알파벳(장음, 탁음, 반탁음) 컨트롤버튼으로만 사용되었고, 본 발명의 2차원 컨트롤처리에서는 후속알파벳 입력후 후속컨트롤버튼이 다시 변형알파벳컨트롤버튼으로 사용된 것이다. 이렇게 주용도의 컨트롤버튼(예에서 후속컨트롤버튼)을 다시 변형알파벳컨트롤버튼으로 활용하는 것을 선출원의 2차원 컨트롤처리방법과 구별하기 위하여 편의상 "2차원 재활용컨트롤처리방법" 혹은 "2차원 다용도컨트롤처리방법"이라 부르기로 한다.

02> 선출원의 방법은 입력방법이 단순하고 일관성 있는 장점이 있으나, 상대적으로 입력타수가 약간 더 많아지는 단점이 있다. 본 발명과의 차이을 쉽게 보이기 위하여 선출원에서 컨트롤버튼을 각각 후속컨트롤(2nd, 3rd, 4th, 5th)버튼으로만 그리고 변형알파벳(장음, 탁음, 반탁음) 컨트롤버튼으로만 사용하였던 경우를 그래프로 나타내면 도2-4와 같다. 도2-3, 도2-4에서도 흐린색으로 표시된 부분을 확장하여 적용할 수 있다.

03> 위에서 교차 컨트롤처리방법을 적용함에 있어서, 컨트롤 후입력의 예를 보였는데, 이 역시 선출원에서 컨트롤처리방법을 적용함에 있어서 컨트롤 선입력 혹은 후입력을 적용할 수 있

는 것과 마찬가지로, 컨트롤 선입력에 의하여 입력할 수도 있다. 예를 들어 い = "반대편 컨트롤버튼" + い = [#] + [*]+あ 와 같이 된다. 그러나 교차 컨트롤처리방법을 적용함에 있어서는 컨트롤을 후입력하는 것이 편리할 것이다.

14> 이는 일본어의 경우뿐만 아니라 다른 모든 언어에 있어서도 동일하게 적용될 수 있다. 예를 들어, 아랍어에서 대표알파벳제외 컨트롤처리방법을 적용하여, 후속알파벳을 입력하기 위하여 [*]과 [#]이 사용되는 경우, 첨자형태의 모음을 입력하기 위하여 다차원 교차컨트롤처리방법을 사용할 수 있다. 또한 태국어에서 [*] 버튼과 [#] 버튼을 각각 후속자음과 후속모음을 입력하기 위한 컨트롤버튼으로 활용하는 경우에도, 성조부호를 입력하는데, 다차원 교차컨트롤처리방법을 사용할 수 있는 것이다.

05> 4. 한국어에 있어서 모음요소를 활용하는 방법

06> 4.1 격음과 경음을 컨트롤처리하는 방법

07> 한국어의 모음요소(ㅡ, ㅣ, .)을 이용할 수 있다. 도4-5를 참고한다. 본 발명에서는 10개의 기본자음 중 9개의 기본자음을 [1] ~ [9] 각 숫자 버튼에 배정하고, 임으로 배정하 키패드상 맨 하단의 [*] 및 [#] 버튼에 한국어의 모음요소 "ㅡ", "ㅣ" 를 고, [0] 버튼에 모음요소 "."을 배정한다. 다양한 방법의 배정이 가능하나 편의상 본 발명에서는 도4-5의 배정 및 배치를 예시한다. 배치된 기본자음과 모음요소는 당연히 기준격자의 위치에 배치되는 것으로 할 수 있고, 배치된 기본자음과 모음요소의 선택에 부분전체선택방법을 적용할 필요는 없다.

3> 격음과 경음을 컨트롤처리함에 있어서, "격음컨트롤" 및 "격음컨트롤"을 "ㅡ" 혹은 "ㅣ"가 배치된 버튼에 각각 임의로 배정하고, 나머지 격자중 기준격자에 가까운 격자에 배치한다. 즉 컨트롤이 배정된 버튼 2타에 격음컨트롤 및 경음컨트롤이 선택되도록 하는 것이다. 격음컨트롤(혹은 경음컨트롤)이 배정된 버튼에 경음컨트롤(혹은 격음컨트롤)을 추가로 배치하여, 배치된 버튼 3타로 선택하도록 할 수도 있다. 선출원에서 설명한 바와 같이 "ㅅ"은 격음이 없으므로 "ㅆ"을 경음 및 격음으로 간주할 수 있다. 또한 격음컨트롤 혹은 경으컨트롤이 반드시 버튼상에 표시될 필요는 없으며 사용자는 그 입력규칙을 숙지하고 있으면 된다.

9> 격음컨트롤을 선택한 상태를 편의상 "{격음}" 으로 나타내고, 도4-5의 예에서 컨틀롤 선입력을 적용하면, ㅋ = {격음}+ㄱ = [*]+[*]+[1]이 된다. 마찬가지로 ㄲ = {경음}+ㄱ = [#]+[#]+[1]이 된다. 경음컨트롤을 격음컨트롤이 배정된 버튼 3타로 선택하는 것을 허용하면, ㄲ = {경음}+ㄱ = [*]+[*]+[*]+[1] 이 된다.

10> 본 발명에서 경음컨트롤과 격음컨트롤을 모음 알파벳요소 중 ㅡ, ㅣ 와 함께 같은 버튼에 배정하고 반복선택방법을 적용하여도 모호성이 없는 이유는 한국어의 특성상, 모음 "ㅡ"는 단어에서 연속하여 2번 등장하지 않기 때문에, 예에서 격음컨트롤을 선택하기 위하여 [*] 버튼을 2타 입력한 것이 2개의 모음 "ㅡ" + "ㅡ" 을 선택한 것으로 인식되지는 않기 때문이다. 모음 "ㅣ"의 경우도 마찬가지이다. "예의"라는 단어(예에서 [8]+[0]+[0]+[#]+[#]+[0]+[*]+[#]) 에서 모음 "ㅣ"가 2번 등장하나 이 경우는 모음 알파벳요소 "." 에 이어 나오므로 여기에서의 "[#]+[#]"이 경음컨트롤을 선택한 것이 아니라 모음임을 알 수 있다. 왜냐하면 예에서, 모음 알파벳요소 "." 으로 단어가 종료될 수 없으므로 그 뒤에 이어서 입력된 [#]은 모음임을 알 수 있기 때문이다. 모음 "ㅡ"는 연속하여 등장하는 경우가 전혀 없으므로, 모음 "ㅡ"가 배정된 버튼에 격음컨트롤과 경음컨트롤을 배정하더라도 모호성은 결코 없다.

1> 그리고 10개의 기본자음 중 9개의 자음을 9개 버튼([1] ~ [9] 버튼)에 배정하고 배치하였으므로, 나머지 한 개의 기본자음이 버튼상에 배정되지 않았다. 이를 편의상 "9버튼탈락자음"이라고 부르기로 한다. 이 9버튼탈락자음은 발음 혹은 모양의 유사성을 고려하여 10개의 기본자음 중 하나의 기본자음을 변형알파벳 혹은 후속알파벳으로 간주하여 컨트롤처리하는 것을 생각할 수 있다. 예를 들어 "ㄹ"을 버튼상에 배치하지 않고, "ㄹ"을 "ㄴ"의 변형알파벳(격음 혹은 경음)으로 간주하여, 컨트롤 선입력시 ㄹ = [*]+[*]+ㄴ 혹은 ㄹ = [*]+[*]+[*]+ㄴ 혹은 ㄹ = [#]+[#]+ㄴ 으로 처리하는 것이다. 도4-5의 예에서 역시 발음과 모양의 유사성을 고려하여 "ㅎ"을 "ㅇ"의 변형알파벳(격음 혹은 경음)으로 간주하여 ㅎ = [*]+[*]+ㅇ 혹은 ㅎ = [*]+[*]+[*]+ㅇ 혹은 ㅎ = [#]+[#]+ㅇ 으로 처리할 수 있다.

12> 이 경우 컨트롤처리되는 알파벳(도4-5의 예에서 "ㅎ")은 알파벳 입력의 편리를 위해 경음 혹은 격음이 존재하지 않는 알파벳(ㄴ, ㅁ, ㄹ, ㅎ, ㅇ) 중의 하나인 것이 좋을 것이다. 컨트롤처리되는 알파벳의 기본 알파벳(도4-5의 예에서 "ㅇ")역시 경음 혹은 격음이 존재하지 않는 알파벳 중의 하나 이어야 알파벳 입력의 편리성을 더할 수 있음은 당연하다. "ㅎ"을 컨트롤 처리하는 예시를 제시한 이유는 경음, 격음이 존재하지 않고, 경음, 격음이 존재하지 않는 알파벳 중 사용빈도 또한 가장 낮은 알파벳 이기 때문이다. 또한 발음에 있어서도 유사성이 있으며, 모양에 있어서도 다른 평음과 격음의 관계처럼 유사성이 있기 때문이다.

113> 여기서 한국어가 전화번호 및 각종 코드의 암기용으로 쓰이기 위해서는 10개의 기본자음(평음)이 10개의 숫자 버튼에 배치되는 것이 바람직하다. 따라서 "ㅎ"을 "."이 배정된 버튼(도4-5의 예시에서 [0])에 추가로 배정하고 해당 버튼 3타에 "ㅎ"을 선택할 수 있는 격자에 배치할 수 있다. "."은 두번 연속 등장하는 경우가 다수이므로 반복선택방법을 적용하여 모호성이 없으려면, 3타에 선택되도록 하여야 하기 때문이다. 이 경우도 물리적으로 기준격자에서 3

번째로 가까운 격자에 "ㅎ"을 배치하지 않더라도 사용자는 단지 해당 버튼 3타로 "ㅎ"을 선택할 수 있다는 것을 인지하고 있으면 된다. 이는 "ㅎ"이 [0] 버튼 3타에 입력될 수 있다는 의미도 있지만, 기본 자음을 모두 각 숫자버튼에 배치함으로써 각종 암기용으로 유용하게 사용될 수 있다.

14> 도4-5에서 알파벳의 식별성을 높이기 위하여 자음은 파란색으로, 모음 혹은 모음요소는 붉은색으로 표시하였으며, 숫자는 검은색으로 표시하였다. [0] 버튼에 푸른색으로 "—"를 표시한 것은 ". + — + 0(zero)"가 전체적으로 "ㅎ" 모양을 나타내므로 [0] 버튼에 "ㅎ"이 암시적으로 배정되어 있는 것을 나타낸다.

15> 즉 9버튼탈락자음은 특정 기본자음의 변형알파벳으로 간주되어 컨트롤처리방법으로 입력할 수도 있고, 동시에 모음요소 "."이 배정된 버튼 3타로 입력될 수도 있다.

16> 4.2 경음의 기본자음 조합처리 및 경음, 격음의 은닉형/비은닉형 반복선택처리

17> 한국어의 모음요소를 활용한 방법에 있어서도 경음을 기본자음의 조합으로 처리할 수 있다. 더구나 모음요소를 활용한 방법에서는 숫자버튼에 기본자음이 하나씩 배정되므로, 경음을 기본자음의 조합으로 처리하더라도, 현행 2벌식 키보드에서와 마찬가지로 "오뚜기" <=> "오ㄷ두기" 와 같은 경우에만 모호성(풀코드 간의 모호성으로 1차모호성)이 발생한다. ("오ㄷ"에서 "ㄷ"은 종성임)

18> 그러나 역시 모호성 발생시 이를 회피할 수 있도록 "색인"을 구비하는 것으로 해결할 수 있다. 결국 경음을 기본자음의 조합으로 입력하는 경우, 격음만을 컨트롤처리방법에 의하여 입력하면 되므로 입력규칙을 더욱 단순화할 수 있다. 격음컨트롤을 선출원에서 설명한 바와

같이 적절한 격자에 둘 수 있다. 예를 들어 도4-1 혹은 도4-2에서 경음컨트롤을 제거하는 것이다.

<119> 또한 기본자음, 격음, 경음의 순으로(예를 들어 ㄱ, ㅋ, ㄲ) 해당 버튼의 누름횟수에 따라 입력(예를 들어 ㄱ=[1], ㅋ=[1]+[1], ㄲ=[1]+[1]+[1])하도록 할 수 있음을 밝힌 바 있다. 이 경우도 선출원에서 밝힌 대로 1차모호성이 발생할 수 있다. 모호성이 발생하는 구체적인 경우는 여기에서 소개하지 않는다. 이 경우 기본자음, 격음, 경음을 버튼상에 모두 표시할 수도 있고 기본자음만을 표시할 수도 있다고 하였다. 이는 후속알파벳(예를 들어 ㅋ, ㄲ)을 반복선택방법에 의하여 선택하는 것이다.

<120> 여기서 대표알파벳만을 표시하고 나머지 후속알파벳을 반복선택방법에 의하여 선택하는 방법을 편의상 "은닉형 반복선택처리방법"이라 하기로 한다. 다시 경음과 격음의 부속순위를 기본자음-경음-격음의 순으로 둘 수도 있다. 예를 들어, ㄱ(1st:기준격자위치), ㄲ(2nd), ㅋ(3rd) 와 같이 두는 것이다. 즉 ㄱ = [1], ㄲ = [1]+[1], ㅋ = [1]+[1]+[1] 로 된다. 이는 쌍자음을 기본자음의 조합으로 처리(혹은 은닉형 반복선택처리)하고 격음까지도 은닉형 반복선택처리방법에 의하여 선택하는 것으로 볼 수 있다.

<121> 이경우 선출원에서 제시한 은닉형 반복선택처리방법과 후속알파벳의 부속순위만 다르나, 경음입력시 숫자버튼 2타(기본자음의 조합)로 입력할 수 있도록 함으로써, 사용자 입장에서의 자연스러움을 배가시킬 수 있다.

<122> 또한 "ㅎ"을 입력시 [0] 버튼 3타로 입력하였는데, "ㅇ"과 "ㅎ"간의 관계는 모양 및 발음상 평음(기본자음)과 격음의 관계와 유사하고 격음역시 숫자버튼 3타로 입력할 수 있으므로 전체적으로 사용방법의 일관성을 높일 수 있게 된다.

▷ 여기서 "ㅎ"을 입력함에 있어서 "ㅎ" 이 배정된 [0] 버튼 3타로 입력하는 것을 허용하면서, 동시에 "ㅎ"을 "ㅇ"의 변형알파벳으로 간주하여 컨트롤처리방법에 의하여 입력하는 것도 가능한 것처럼, 격음을 해당 기본자음이 속한 버튼 3타(혹은 2타)로 입력할 수 있도록 하면서, 동시에 컨트롤처리방법에 의하여 입력하는 것을 얼마든지 허용할 수 있음은 물론이다. 사용자가 컨트롤처리방법에 의하여 격음을 입력하면, 시스템은 격음입력시 모호성(풀코드간의 모호성으로 즉, 1차모호성) 없이 격음을 식별할 수 있고, 해당 기본자음이 속한 버튼 3타로 입력하면 경우에 따라 모호성이 발생할 수도 있다. 이는 경음에 대해서도 마찬가지이다.

4> 즉 동일한 키패드에서 사용자는 격음입력시, 컨트롤처리방법에 의하여 입력할 수도 있고, 은닉형 반복선택방법에 의하여 입력할 수도 있는 것이며, 더불어 경음에 대해서도 마찬가지이다. 예를 들어 격음컨트롤을 [*] 버튼 2타로 선택하도록 하고, 동시에 경음 및 격음은 기본자음이 속한 버튼 2타 및 3타로 선택하도록 하여, "카키"를 입력시, "카키 = [1]+[*]+[*]+[#]+[0] + [1]+[1]+[1]+[#]" 로 할 수 있는 것이다. 즉 첫번째 "ㅋ"은 컨트롤처리방법으로 입력하고, 두번째 "ㅋ"은 반복선택방법으로 입력할 수 있는 것이다. 마찬가지로 "까끼"에 대해서도 "까끼 = [1]+[#]+[#]+[#]+[0] + [1]+[1]+[#]" 로 할 수 있는 것이다.

25> 경음과 격음을 비은닉형 반복선택처리방법 혹은 은닉형 반복선택처리방법에 의하여 입력시 경음과 격음의 선택순서 역시 사용자의 편리대로 설정하도록 할 수 있다.

26> 4.3 한국어의 4개 모음요소를 활용하는 방법

27> 도4-5의 경우를 편의상 "10자음 3모음법" 혹은 간단히 "3모음법" 이라 부르기로 한다. 도4-5의 사례를 풀어 도4-6에서와 같이 표현할 수 있다. 도4-6에서도 도4-5에서와 동일한 입력방법을 적용할 수 있는데, 도4-6에서 보듯이 이는 한국어의 모음 "ㅕ" 혹은 "ㅛ" 등을 입력시 필요한 모음요소 ":(경우에 따라 수평방향 혹은 수직방향의 획으로 작용하나 여기서는 편의상 수평방향의 획 ':' 으로 표시)" 를 모음요소 "." 과 함께 그룹핑하여 같은 버튼에 배정하고, 모음요소 "." , ":" 을 각각 반복선택방법에 의하여 배정된 버튼 1타, 2타로 각각 선택하되, 모음요소 ":" 을 명시적으로 버튼상에 배치하지 않는 것으로 볼 수 있다. 즉 도4-5의 경우 역시 도4-6에서 보듯이 4개의 모음요소("ㅡ" , "." , ".." , "ㅣ")를 활용하는 것으로 볼 수 있으며, 이를 편의상 "점(點)모음요소 반복선택 4모음법" 이라 한다.

128> 도4-7은 모음요소 ":" 을 모음요소 "." 과 함께 그룹핑하지 않고, 각기 다른 그룹으로 그룹핑한 경우를 보여준다. 즉 4개의 모음요소("ㅡ" , "." , ":" , "ㅣ")가 각각 다른 버튼에 배정되며, 이를 편의상 "10자음 4모음법" 혹은 "4모음법" 이라 부르기로 한다. 4모음법은 3모음법과 통하는 점이 많으므로 비교적 간단히 설명하며, 3모음법에서 설명한 내용중 4모음법에 적용할 수 있는 내용은 4모음법에서 그대로 적용될 수 있는 것은 자명하다. 용어 역시 3모음법에서 사용한 용어를 유사하게 사용할 수 있다.

129> 4모음법에서 10개의 기본자음 중 격음과 경음이 존재하지 않는 2개의 기본자음을 제외한 나머지 자음을 8개의 버튼에 각각 배정하고, 4개의 모음요소 하나하나와 격음컨트롤, 경음컨트롤, 8개의 버튼에 배정되지 않은 2개의 자음(편의상 "8버튼 탈락자음" 혹은 "10-8자음" 혹은 더 간단히 "-8자음" 이라 부름)을 각각 2개씩 그룹핑하여 모음요소는 배정된 버튼 1타로

입력하고, 함께 배정된 컨트롤 혹은 탈락자음들은 배정된 버튼 2타로 입력(즉, 반복선택방법에 의하여 입력)할 수 있다.

0> 모음요소 "." 을 모양이 유사한 [0] 버튼에 배정하고, 숫자 "0" 안에 표시함으로써 버튼상 간결한 배치를 가능하게 할 수 있으며(혹은 숫자안의 표시를 생략해도 모양이 유사하므로 사용자는 이를 쉽게 인지할 수 있음), 모음요소 ":" 을 모양이 유사한 [8] 버튼에 배정하고, 숫자 "8" 내부에 표시함으로써(혹은 숫자안의 표시를 생략해도 모양이 유사하므로 사용자는 이를 쉽게 인지할 수 있음) 버튼당 외형적으로 단 하나의 알파벳만을 배치할 수 있는 장점이 있다. 도4-5의 경우에도 모음요소 "." 을 [0] 버튼의 숫자 "0" 안에 표시하고 "ㅎ" 을 명시적으로 표시할 수도 있다. 도4-7에서와 같이 모음 "ㅡ", "ㅣ" 가 배정되는 버튼이 [*] 버튼 및 [#] 버튼인 경우 한국어의 다양한 모음을 입력시 인접한 버튼을 누르게 되어 운지거리를 적게하는 효과가 있다.

31> 4모음법의 경우도 격음과 경음의 입력에 있어서, 3모음법의 경우와 마찬가지로 반복선택방법을 적용할 수 있으며, 이를 통칭하여 편의상 "격음/경음 반복선택 4모음법" 이라 부른다. 또한 3모음법에서의 사례와 같이 반복선택방법을 혼용하여 적용할 수 있으며, 이를 통칭하여 편의상 "격음/경음 컨트롤처리 반복선택 혼용 4모음법" 이라 한다.

32> 도4-7의 키패드 역시 반복선택방법 관점에서 도4-8과 같이 풀어 표현할 수 있다. 도4-6과 도4-8를 비교해 보면, 도4-5와 도4-7의 차이를 명확히 알 수 있게된다. 도4-8은 도4-6에서 [0]버튼 2타로 선택하던 모음요소 ":" 을 [8]버튼으로 옮겨 1타로 입력하도록 한 것과 같은 결과이다.

3> 모음요소 "." 과 ":" 을 [*] 버튼 및 [#] 버튼에 배정할 수도 있으나, 이 경우는 도 4-7에서와 같이 모음요소를 숫자 안에 표시함으로써, 표면적으로 단 하나의 알파벳만 배치할 수 없게 된다.

4> 도4-7을 기준으로 입력예를 들면 ㅕ = : + ㅣ = [8]+[#], ㅏ = ㅣ + : = [#]+[8], ㅠ = ㅡ + : = [*]+[8] 이 되며, 8버튼 탈락자음을 반복선택방법에 의하여 입력시, ㅇ = [8]+[8], ㅎ = [8]+[8] 이 된다. 만약 모음요소 ":" 을 모음요소 "." 이 배치된 버튼 2타로 입력하는 것을 허용하면, "ㅎ" 은 3모음법원에서와 마찬가지로 "." 이 배정된 버튼 3타로 입력하도록 하면 된다.

5> 입력예로, "워" 혹은 "웨" 의 경우 모음 "ㅡ"와 "ㅣ" 사이의 점모양의 모음요소 ".."은 모음요소 ":"으로 입력하면 된다. 시스템에서 일시적으로 "유 + ..." 를 입력한 것으로 인식될 수 있으나 "유" 다음에 모음 "ㅣ"가 나올 수 없으므로 시스템은 "워"를 입력한 것으로 인식할 수 있다. "10자음 점(點)모음요소 반복선택 3모음법" 에서도 "워" 의 경우 "ㅡ" 와 "ㅣ" 사이의 점모양의 모음요소를 입력시 모음요소 "." 을 두번 입력하는 것으로 혹은 모음요소 ":" 을 한번 입력하는 것으로 이해할 수 있다.

6> 3모음법과 4모음법은 한국어의 자음과 모음의 출현규칙(즉, 단어생성규칙, 알파벳 결합규칙)을 이용하여 "언어제한 반복선택방법" 중에서도 "한국어제한 반복선택방법" 에 최적화된 한국어 입력방법을 제시한 것으로 볼 수 있다.

7> 4.4 완전컨트롤처리방법

▷ 도4-5에서 격음과 경음을 컨트롤처리방법에 의하여 입력하더라도, "이허 <=> 아히", "이혀 <=> 아허 <=> 야히" 와 같은 경우에 모호성이 발생할 수 있다(양자의 풀코드 값은 모두 같으므로). 도4-7에서 "이며 <=> 야미" 와 같은 경우가 있을 수 있다.

▷ 3모음법에서와 "ㅎ"을 "ㅇ"의 변형알파벳으로 간주하여 컨트롤처리방법에 의하여 입력(예. ㅎ = ㅇ+{격음} = ㅇ+[*]+[*])할 수 있는 것과 마찬가지로 2개의 8버튼 탈락자음을 컨트롤처리방법에 의하여 입력로 입력할 수 있다. 이를 편의상 "완전컨트롤처리방법" 이라 부른다. 더 나아가 탈락자음(예. "ㅎ")이 배정된 버튼과 "—" 와 함께 배정된 컨트롤을 조합하여 탈락자음을 입력할 수 있다. 예를 들어 "ㅎ = . + {격음} = [0]+[*]+[*]" 로 입력하는 것이다. 4모음법에서의 8버튼 탈락자음 역시 유사하게 입력할 수 있다. 이를 편의상 "탈락자음배정버튼이용 완전컨트롤치리방법" 이라 부른다.

5. 색인을 통하여 모호성을 극복하는 방법

반복선택방법 적용시 여러 단어가 하나의 코드로 표현되어 모호성이 발생할 수 있다. 도4-1의 입력예에서 "고이"가 "굉"으로 "굉"이 "고이"로 인식될 수 있는 모호성이 있음을 지적했다. 그런데 보통은 단어단위로 단어를 입력하는데, 예를 들어 "굉장히" 라는 단어는 존재하지만 "고이장이"라는 단어는 존재하지 않는다. 도4-1에서 양자는 모두 같은 코드값을 갖는다. 따라서 이에 대하여 단말기(클라이언트) 혹은 서버측에 특정 키패드와 특정 알파벳입력방법에 대하여 모호성이 발생할 수 있는 단어들에 대한 색인을 준비하고 이를 등록해 둔다. "굉장히" 혹은 "고이장이"가 입력이 되었을 때 시간지연 등에 의해서도 양자를 구별할 수 없을 때, 색인

에 "꽹장히"가 올바른 단어로 등록이 되어 있다면, 사용자가 입력을 원하는 단어(이하에서 편의상 "타겟단어"라고 함)는 "꽹장히"인 것을 알 수 있는 것이다. 이는 시스템이 구별할 수 없는 모호성이 발생했을 경우만 적용되며, 시스템(단말기측 혹은 서버측 시스템)에서 정의한 시간지연값 혹은 사용자가 정의한 시간지연값에 의하여 구별이 가능한 경우는 그에 따라야 함은 당연하다. 즉 사용자가 의도적으로 끊어(시간지연 혹은 공백 혹은 우측진행 버튼 혹은 기타의 특정 버튼을 사용하여) "고이장이"를 입력하였다면 비록 의미상 그른 단어라 할지라도 그 단어가 타겟단어이므로 "고이장이"를 출력하여야 한다.

<42> 색인을 준비하는 방법은 모호성이 발생할 수 있는 경우의 단어에 대하여 올바른 단어(예에서 "꽹장히")만을 등록해 둘 수도 있고, 그른 단어(예에서 "고이장이")만을 등록해 둘 수도 있다. 또한 올바른 단어와 그른 단어를 모두 등록해 두어도 무방하며, 단지 시스템은 색인의 어떤 단어가 올바른 단어인지 그른 단어인지에 대한 정보를 가지고 있으면 된다.

<43> 이는 키패드에서 반복선택방법을 적용하는 모든 경우에 적용될 수 있다. 예를 들어 한국의 삼성전자사의 입력방법에서 "국가 <=> 구카" 와 같은 모호성이 발생하는데 여기에도 마찬가지로 적용될 수 있다. 뿐만아니라 도4-1와 유사하게 자모의 쌍을 각 버튼에 배치하고 자음1타, 격음2타, 경음3타, 모음1타로 선택할 수 있게 하면서, 음절(글자) 단위로 음절완성 버튼을 누르는 경우에도 적용될 수 있다. 사용자가 "꽹장히"를 입력시, 일시적으로 "고이장이"가 출력되더라도, 단어종료되는 순간(예를 들어 공백입력 등)에 색인을 참조하여 이를 "꽹장히"로 수정할 수 있다.

4> 이때, 단어의 색인이 클라이언트측과 서버측에 모두 존재할 경우 1차적으로 클라이언트측의 색인을 검색하여 올바른 단어의 식별을 시도하고, 식별실패시 2차적으로 서버측의 색인을 검색하여 최종적으로 올바른 단어를 식별하도록 할 수 있다.

:5> 단어의 단위를 구별하는 것은 누구나 생각할 수 있듯이 단어와 단어를 구분해 주는 요소에 의해서 식별하면 된다. 예를 들어 공백과 공백사이, 단어시작과 공백사이, 공백과 입력끝사이, 공백과 모드전환 사이 등등의 경우를 생각할 수 있다. 색인을 참조하여 올바른 단어를 결정하는 것은 단어 단위로 이루어지므로 단어의 구분을 주는 요소 입력시 색인참조가 이루어지고 올바른 단어가 결정될 수 있다.

16> 이는 반복선택방법을 적용하여 모호성이 발생하는 모든 경우에 적용될 수 있다. 예를 들어 현재 한국에서 사용되고 있는 삼성전자社의 방법에서 종성과 초성이 같은 버튼에 속한 자음인 경우 모호성이 발생할 수 있다. "국가" <=> "구카" 를 예로 들 수 있다. 알파벳 별로 코드를 두어 입력시, 음절(글자) 단위를 인식하지 않고 입력하므로 사실상 많은 경우에 무의식적으로 "국가"를 입력하면 "구카"로 된다. 이 경우도 시간지연 등에 의하여도 두 단어중의 어느 단어가 타겟단어인지를 알 수 없으면, 시스템(단말기 혹은 서버)은 색인을 참조하여 타겟단어를 알아낼 수 있다. 단 이 경우는 "국"의 종성 "ㄱ"과 "가"의 초성 "ㄱ"이 연속하여 눌러졌다고 하더라도, 대부분의 경우 타겟단어는 "국가"인 경우가 많으므로, 시간지연에 상관없이 색인을 참조하여 타겟단어를 식별해 내는 방법도 효율적일 수 있다.

:47> 또한 이는 한국어에만 국한되지 않고 반복선택방법을 사용하는 경우의 모든 언어에 적용될 수 있는 것임은 자명하다. 예를 들어 영어입력에 있어서 혹은 다른 언어 키패드의 영어모드에서, [2]-[2]-[2]를 입력한 것이 "AB" 인지 "BA" 인지 알 수 없는 모호성이 발생한 경우에 "AB" 가 올바른 단어로 색인에 등록이 되어 있다면, 이를 인식하여 사용자에게 타겟단어로 제

공할 수 있는 것이다. 만약 모호성이 발생하는 복수의 단어가 모두 올바른 단어로 색인에 등록이 되어 있다면, 사용빈도가 많은 단어를 사용자에 1차적으로 제공하고 사용자로 하여금 최종적으로 타겟단어를 확정하도록 할 수 있다.

<8> 만약 어떤 단어의 입력에 대하여 모호성이 있을 경우, 색인을 참조하여도 그 입력에 해당하는 단어들이 모두 올바른 단어로 인식된다면, 사용자에게 적절한 방법(시각적, 청각적)에 의한 피드백을 주어 선택하게 하면 된다.

<19> 사용자로 하여금 타겟단어를 확정하게 하는 방법은 디스플레이창에 올바른 단어로 인식된 복수의 단어를 사용빈도순서(혹은 우선순위순서)로 나열하고 상하 스크롤에 의하여 혹은 디스플레이된 순서에 해당하는 숫자버튼을 이용하여 사용자로 하여금 타겟단어를 확정하도록 할 수 있다. 다른 방법으로는 사용빈도가 높은 단어만을 표시하고 타겟단어가 아닌 경우 다음 단어를 표시되도록 할 수 있는 컨트롤(편의상 "다음단어컨트롤"이라 함)을 선택함으로써 다음으로 사용빈도가 많은 단어를 표시되도록 하고, 역시 타겟단어가 아닌 경우 마찬가지로 다음단어컨트롤을 다시 선택함으로써 계속해서 타겟단어를 검색하도록 할 수 있다. 이 경우 타겟단어를 검색한 후 임의의 다른 버튼의 입력(예를 들어 다른 알파벳의 선택 혹은 공백, 모드전환 등 다음단어컨트롤 선택의 선택이 아닌 일체의 버튼 입력)으로 타겟단어가 확정되도록 할 수 있다.

<50> 다음단어컨트롤의 선택은 선출원에서 기술한 "부분전체선택방법" 혹은 "반복선택방법(기준/단순)" 모두 적용될 수 있다. 특정 버튼의 기준격자의 위치에 다음단어컨트롤이 배치되어 있는 것으로 한다면, 그 특정 버튼 1타로 다음단어컨트롤이 선택될 것이다.

1> 6. 심플코드 활용방법 및 단축입력방법/병행입력방법

2> 6.1 알파벳 연관 심플코드 생성방법

3> 정보통신 단말기를 사용하여 정보시스템에 접근할 경우 알파벳의 입력이 필수적이다. 혹은 입력하여야 할 알파벳을 숫자코드화 하여 입력할 수 있다. 소형화된 정보통신 단말기에서는 보통 키패드 형태의 인터페이스를 채용하고 있는 경우가 많다. 여기서 말하는 코드란 모든 종류의 코드를 말하는 것으로 대표적인 예를 들면 전화번호, 증권종목(상장회사)코드, 도시코드, 부서코드, 전철역코드, 은행코드 등 헤아릴 수 없이 많다. 각종의 명칭을 코드화하여 입력하는 잇점은 입력을 단순화 할 수 있다는 것이다.

54> 정보통신 단말기란 PC, 이동통신기기, 스마트폰, PDA, 양방향 문자 송수신 기기, ATM기(자동 입출금기) 등 모든 형태의 정보통신 단말기를 포함한다. 뿐만 아니라 전자수첩과 같이 통신기능이 없는 단말기까지 망라한다. 정보시스템이란 GUI통하여 시각적으로 접근할 수 있는 시스템뿐만 아니라 ARS와 같이 소리로만 접할 수 있는 시스템 등 모든 형태의 시스템을 포함한다. 또한 시스템은 좁게는 서버측 시스템을 의미하며, 넓게는 서버 시스템과 호응하는 단말기의 클라이언트 소프트웨어도 포함한다.

55> 키패드상 배치된 알파벳을 이용하여 각종 코드의 암기용으로 유용하게 사용할 수 있다. 코드의 암기용으로 사용함에 있어서, 심플네이밍, 이니셜네이밍, 풀네이밍에 의한 방법이 있을 수 있다. 그 내용을 간단히 요약하면 다음과 같다.

6> 심플네이밍은 코드화 하고자 하는 단어 혹은 구절(단어 혹은 구절을 통칭하여 이하에서 간단히 "어구"라 함)과 연관된 숫자를 코드로 지정하는 것이다. 예를 들어, 한국어의 경우, 기업명코드의 경우, 도4-2을 기준으로 "가산전자"의 코드를 ㄱ, ㅅ, ㅈ, ㅈ 과 연관된 "1799"를 심플코드로 하는 것이다. 이 경우 "가산전자"에서 심플코드 "1799"와 연관된 ㄱ, ㅅ, ㅈ, ㅈ 을 부각하여 표시함으로써 특정 단어의 심플코드를 더욱 쉽게 알 수 있도록 할 수 있으며, 더 나아가 그러한 어구로 부터 심플코드를 추출해 낼 수도 있다. 심플네이밍의 경우 코드화 하고자 하는 단어에 속한 어떤 알파벳과 연관된 숫자를 코드로 정하는 것이므로, 예로 든 "가산전자"의 코드를 반드시 "1799"로 할 필요는 없다. 예를 들어 "주식이란"의 심플코드를 ㅈ, ㅅ, ㅣ, ㄹ 과 연관된 값을 심플코드로 할 수 있듯이, 가산전자의 경우 다른 예를 들면, ㄱ, ㅅ, ㄴ, ㅈ 과 연관된 "1729"로 할 수도 있고, ㄱ, ㅅ, ㅓ, ㅈ 과 연관된 "1749"로 할 수도 있고, "가산전자"를 구성하는 전체 알파벳과 연관된 "13294293"으로 할 수도 있다. 어구를 구성하는 전체 알파벳과 연관된 코드를 심플코드로 하는 것을 편의상 "전체연관 심플네이밍"이라고 하고 특정 어구의 일부 알파벳과 연관하여 심플코드를 지정하는 것을 편의상 "일부연관 심플네이밍"이라고 부른다. 어떤 경우이든 심플네이밍(즉 심플코드)은 어구단위로 그 어구를 구성하는 알파벳에 연관되어 코드를 지정하는 것으로 볼 수 있다. 이는 한국어 이외의 언어에도 마찬가지로 적용할 수 있다. 예를 들어 "captain"에서 자음 "CPTN" 과 연관되는 "2786"를 심플코드로 할 수 있는데 이는 일부연관 심플코드이며, 이를 특히 편의상 "자음연관 심플코드"라고 부르기로 한다.

157> "escape"와 같이 모음으로 시작하는 단어에 대하여 자음연관 심플코드를 생성하면, "SCP"에 해당하는 "727"이 되는데, 이는 "scape"의 자음연관 심플코드와 동일하게 된다. 따라서 축약된 심플코드를 사용하되, 정의되는 심플코드와 심플코드 대응어구의 중복을 최소화하고, 사

용상의 편리를 더하기 위하여 첫모음과 자음을 연관한 심플코드를 생각할 수 있다. 이를 편의상 "첫모음+자음기준 심플코드"라 이름붙인다. 이러한 첫모음+자음연관 심플코드 역시 다른 심플코드와 마찬가지로 특정 단어에 대하여 심플코드를 기계적으로 생성해 낼 수 있는 장점이 있다.

<58> 단어뿐 아니라 구절 역시 심플네이밍에 의하여 코드화될 수 있다. 선출원에서 예를 든 "어디로 가려는가"라는 어구에서 의미를 가장 잘 함축하고 있는 글자(음절)을 활용하여 ㅇ, ㄷ, ㄱ, ㄹ 에 매핑되는 "8314"를 심플코드로 할 수 있다. 영어의 경우 선출원에서 예를 든 "date tonight" 에서 음가를 가지고 있는 d, t, t, n 에 연관되는 "3886"을 심플코드로 할 수 있다.

<59> 이니셜네이밍은 일부연관 심플네이밍의 특별한 경우이다. 한국어의 경우 이니셜네이밍은 음절(글자)을 기준으로 초성(첫자음)에 매핑되는 숫자를 코드로 지정하는 방법이 있을 수 있다. 이를 편의상 "음절(글자)기준 이니셜네이밍"이라고 한다. "가산전자"의 경우 음절기준 이니셜네이밍에 의한 이니셜코드는 각 음절(글자)의 자음초성에 연관된 "1799"가 된다. 음절기준 이니셜네이밍의 경우도 한국어 이외의 언어에 대해서도 적용할 수 있다. 예를 들어 영어의 경우, "entertainment" 라는 단어의 음절기준 이니셜네이밍에 의한 이니셜코드를 e, t, t, m 에 연관되는 "3886"으로 할 수 있다. 음절기준 이니셜네이밍은 한 음절이 하나의 글자를 이루는 한국어의 경우에 더욱 유용하게 사용될 수 있다. 한국어 뿐만 아니라 한국어와 마찬가지로 한 음절이 하나의 글자를 이루는 중국어, 일본어 등의 다른 언어에 있어서도 유용하게 사용될 수 있다. 중국어의 경우 北京(Beijing : 첫번째 'e' 에는 4성조부호가, 마지막 'i' 에는 2성조부호가 붙음)의 음절기준 이니셜코드는 도1-1을 기준으로 b, j 에 연관되는 "25" 가 되며, 도10-1 ~ 10-4 를 기준으로는 "14" 가 된다.

0> 마찬가지로 구절에 대해서도 이니셜 네이밍이 가능한데, 선출원에서 예를 든 "어디로 가려는가" 라는 구절에서 각 단어의 첫 알파벳인 ㅇ, ㄱ 과 연관된 "81"을 이니셜 코드로 할 수 있다. 영어의 경우 선출원에서 예를 든 "dance with the wolf" 라는 어구에 대하여 단어기준 이니셜네이밍의 이니셜코드는 d, w, t, w 와 연관된 "3979"가 이니셜코드가 된다. 단어기준 이니셜네이밍은 구절 전체에 대하여 코드를 부여할 경우 모든 언어에서 유용하게 사용될 수 있다.

61> 심플코드(전체연관 심플코드 부분연관 심플코드)와 이니셜코드(음절기준 이니셜코드, 단어기준 이니셜코드)를 통틀어 편의상 "심플코드(즉 넓은 의미의 심플코드)" 혹은 "단축코드"라고 부르기로 한다. 전체연관 심플코드, 자음연관 심플코드, 음절기준 이니셜코드, 단어기준 이니셜코드는 특히 그 코드 생성에 있어서 규칙성이 있으므로, 이를 범용하게 사용할 수 있으며, 타인이 그러한 규칙으로 심플코드를 생성해 놓은 경우에도 쉽게 사용할 수 있다.

62> 풀네이밍은 특정 문자입력방법에 의해 코드화하고자 하는 어구의 입력값이다. 따라서 이는 적용되는 문자입력방법에 따라 달라질 수 있으며, 알파벳단위로 어떤 어구에 해당하는 숫자값을 코드로 하는 것으로 볼 수 있다. 선출원에서 예를 든 "서울"의 경우 도4-2을 기준으로 부분전체선택방법(선출원에서 언급)에 의한 풀코드는 "7745888944"가 된다. 만약 기준반복선택방법(선출원에서 언급)에 의한 풀코드는 "7448884" 가 된다. 만약 도4-2와 다른 키패드를 적용하거나 또 다른 알파벳 입력방법을 적용할 경우는 그에 따른 풀코드 값이 있게 될 것이다.

163> 6.2 기존의 중국어 한자 입력방법

64> 중국어의 입력 방법은 한글에서의 한자 입력방법과 같이, 중국어 발음에 해당하는 영문을 입력하고 "한자 변환키"를 사용하여 변환가능한 한자가 되시되면 해당하는 한자를 선택하여 입력하는 것이 일반적인 형태이다. 즉, 로마알파벳으로 한어병음을 입력하면, 시스템에서 이에 대응되는 한자를 검색하여 사용자에게 제공하는 방식이었다. 도5-1을 참고한다. 중국어에 있어서 풀코드는 중국어의 영어발음을 기준으로 정할 수 있다.

65> 6.3 유니크化 심플코드

66> 심플코드의 해독이 클라이언트측에서 이루어질 수 있는 경우(즉, 클라이언트측에서 특정어구와 그 특정어구에 해당하는 심플코드값을 가지고 있는 경우)는, 입력된 심플코드에 대응하는 어구를 서버측에 전달할 수 있다. 클라이언트측에서 심플코드를 해독할 수 있는 경우에도 어플리케이션의 성격에 따라 서버측에서 심플코드를 요하는 경우는 심플코드자체(즉, 숫자나열)를 서버측으로 전송하면 된고 서버측에서 이 심플코드를 해석하면 된다. 이것이 심플코드의 해독이 클라이언트측에서 혹은 서버측에서 이루어질 수 있다고 한 것이다.

67> 다수의 어구에 대하여 심플코드를 정의하여 사용할 경우, 동일한 심플코드에 대응되는 어구가 다수 존재할 수 있다. 이렇게 심플코드간에 발생할 수 있는 모호성을 편의상 "2차모호성"이라 한다. 이 경우 시스템내부에서는 동일한 값을 갖는 심플코드에 일련번호를 덧붙여 유일한 코드값을 만들어 저장할 수 있을 것이나, 사용자는 주로 특정 어구와 연관된 심플코드를 사용하게 되므로 2차모호성이 발생하게 된다. 이러한 경우, 그러한 어구들을 사용 우선순위에 따라 시스템이 사용자에게 추천하여 줄 수 있어야 함은 당연하다. 다른 어구이면서 동일한 심플코드를 갖는 경우, 시스템내부에서 심플코드에 어구의 사용빈도에 따른 우선순위에 따라 일련번호를 붙이므로써, 일련번호를 사용자에게 추천하여 주는 우선순위로 활용할 수 있다. 물

론 반드시 일련번호를 붙여야만 하는 것은 아니고, 우선순위 정보를 시스템이 별도로 가지고 있어도 되며, 단지 그렇게 할 수 있는 예를 보인 것이다.

8> 예를 들어, "증권정보"의 심플코드를 음절기준이니셜코드인 "9196" 과 "주권정보"의 심플코드를 역시 음절기준이니셜코드인 "9196"로 같다면 각 단어의 상용빈도에 따라 일련번호를 붙이고 그 일련번호를 사용자에게 추천하여 주는 우선순위로 사용할 수 있다. "증권정보"의 사용빈도가 높다면 그에 우선순위를 주어 "증권정보" = "91961"으로 하고, "주권정보" = "91962"로 할 수 있다. 마찬가지로 先生(xiansheng : 모음 a 가 제1성조)과 學生(xuesheng : 첫번째 e 가 제2성조)의 음절기준 이니셜코드가 "97" 로써 같다면, 각 단어의 사용빈도에 따라 일련번호를 붙여 그 일련번호를 우선순위로 활용할 수 있다. 先生(xiansheng)의 사용빈도가 學生(xuesheng)의 사용빈도보다 더 높다면, 先生(xiansheng) = 971 로 하고, 學生(xuesheng) = 972 로 할 수 있는 것이다. 이렇게 일련번호를 덧붙여 생성된 심플코드를 편의상 "유니크化 심플코드"라 하기로 하고, 일련번호를 붙이지 않고 중복성이 있는 심플코드를 단순히 "심플코드"라 하며, 양자를 모두 포괄하여 역시 "심플코드"라 한다.

69> 先生(xiansheng)과 學生(xuesheng)의 예에서, 사용자가 "97" 만을 입력하면, 시스템이 先生(xiansheng)과 學生(xuesheng)을 사용자에게 제공하고, 사용자가 이중 선택하도록 할 수 있으며, 사용자가 처음부터 이를 인지하고 "971"을 입력하였다면, 시스템에서는 이를 " 先生(xiansheng)"으로 인식할 수 있는 것이다.

70> 여기서 음절기준 이니셜코드의 기초로 사용된 "x, s" 를 부각하여 표시함으로써, 사용자에게 시각적으로 더 좋은 효과를 줄 수 있다. 부각하여 표시하는 방법 중의 한가지는 先生

(XianSheng)와 같이 대문자로 표시하는 것이다. 더 이렇게 대문자로 표시된 단어로부터 시스템이 심플코드 "97" 을 해석해 낼 수도 있다.

'1> 6.4 심플코드의 활용 사례

'2> 도시이름에 대하여 심플코드(음절기준이니셜코드)를 부여한 사례를 들면 다음과 같다. 이는 철도 정보시스템 등에서 유용하게 쓰일 수 있다.

73> . 서울 = 78, 수원 = 78, 대전 = 39, 신탄진 = 739, . . .

74> 여기서 서울과 수원의 음절기준 이니셜코드가 같으므로, 시스템내에서는 이를 회피하기 위하여 연번을 붙여 "서울 = 781", "수원 = 782" 와 같이 심플코드를 유니크화할 수 있다. 사용자가 시스템에 "78"만을 송출하면 시스템에서는 적절하게 피드백을 주어(예를 들어 서울과 수원의 목록을 제공 혹은 음성으로 알려줌), 사용자로 하여금 서울과 수원중의 하나를 선택하도록 하면된다. 사용자가 처음부터 이를 인지하고 "781"을 입력하였다면, 시스템에서는 이를 "서울"로 인식할 수 있는 것이다.

75> 만약 서버측 시스템에서 심플코드 "78"을 필요로하는 것이 아니고 "서울"이라는 단어 자체를 필요로 하면, 클라이언트측에서 심플코드 "78"을 해석하여 "서울"을 서버측으로 전송하면 된다. 혹은 클라이언트측에서 심플코드를 해석할 수 있는 경우라 하더라도, 애플리케이션의 성격에 따라 서버측에서 심플코드 자체를 필요로하면, 입력된 심플코드 자체를 서버측으로 송출하면 된다.

176> 도시이름에 대하여 심플코드(전체연관 심플코드, 음절기준 이니셜코드, 자음연관 심플코드, 첫모음+자음연관 심플코드)를 부여하는 다른 예를 들면, 도1-1을 기준으로, 北京(Beijing)

의 음절기준 이니셜코드는 b, j 에 연관된 "25" 가 되고, 전체연관 심플코드는 "2345464" 이 되고, 자음연관 심플코드는 b, j, n, g 에 연관된 "2564" 가 된다.

77> 다시 증권종목(상장회사)의 예를 들면 다음과 같다. 이는 각종 증권 정보시스템에서 유용하게 쓰일 수 있다.

78> . 동화제약 = 3098, 디지틀조선 = 39397, 한통프리텔 = 83643, . . .

79> 여기서 예를 들어 "디지털조선"의 경우 음절기준 이니셜코드의 기초로 사용된 "ㄷ, ㅈ, ㅌ, ㅈ, ㅅ"을 부각하여 표시함으로써, 사용자에게 시각적으로 더 좋은 효과를 줄 수 있다.

80> 은행코드의 예를 들면 다음과 같다. 이는 ATM기 및 각종 금융 정보시스템에서 유용하게 쓰일 수 있다.

81> . 국민(은행) = 14, 하나(은행) = 82, . . .

82> 여기서 이렇게 기 정의된 심플코드를 사용자가 입력시 클라이언트(단말기)측에서 해석하여 사용자에게 보여줌으로써 어구입력에 활용할 수 있음은 자명하다. 이를 "단축입력방법"이라 하며, 다음에 "병행입력방법"과 더불어 상세히 설명한다.

83> 6.5 선택빈도에 따른 우선순위 자동 재지정

84> 더 나아가 최초 "증권정보"와 "주권정보" 의 우선순위가 각각 1, 2위로 지정되어 있었으나, 특정 사용자가 "주권정보"를 선택하는 횟수가 현저하게 많다면, "주권정보"의 우선순위를 "증권정보"의 우선순위보다 높은 것으로 조정할 수 있다. 설명한 대로 다양한 방법이 있으나

일련번호를 바꾸어 이를 실현할 수 있다. 우선순위 정보를 별도로 시스템에서 가지고 있는 경우는 그 우선순위 정보를 변경함으로써 가능하다.

5> "주권정보"를 선택하는 횟수가 현저하게 많은지 여부의 판단기준은 시스템이 결정할 수도 있고, 사용자가 (재)지정할 수도 있다. 이러한 판단기준의 예를 들면, 10회 중 8번 이상 기지정된 우선순위와 다르게 선택하면, 기존 우선순위를 자동으로 정정하도록 할 수 있다. 옵션에 따라 사용자에게 정정여부를 물어 확인받을 수도 있다.

36> 이는 다른 언어에 있어서도 동일하게 적용될 수 있으며, 위 사례에서 "주권정보" 와 "증권정보" 를 각각 "先生(xiansheng)" 과 "學生(xuesheng)" 으로 바꾼 경우에도 동일하게 생각할 수 있다.

87> 6.6 심플코드 자동지정 및 심플코드 연관 알파벳의 부각 표시

88> 특히 전체연관 심플코드 이외에 자음연관 심플코드, 음절기준 이니셜코드, 단어기준 이니셜코드는 전술한 바와 같이 심플코드 생성에 있어서 규칙성이 있으므로, 사용자가 특정 어구에 대한 심플코드 지정시, 심플코드 생성규칙을 지정한 상태에서 특정 어구를 입력하면 그에 대응되는 심플코드를 자동으로 추출하여 시스템에 저장할 수 있다. 또한 심플코드와 연관되는 알파벳을 부각하여 표시하여 줌으로써, 사용의 편리성을 높일 수 있다. 영어의 경우 심플코드와 연관되는 알파벳을 부각하여 표시하는데, 대문자를 사용하여 부각하여 표시할 수 있다.

또한 선출원에서 "단축입력방법" 및 "단축/풀 병행입력방법"을 제시한 바 있다. 단축입력을 위한 심플코드는 시스템에서 정의하여 줄 수도 있고, 사용자가 이를 변경할 수도 있고, 또한 사용자는 새로운 어구에 대한 심플코드를 지정할 수도 있다.

새로운 어구에 대한 심플코드를 정함에 있어서, 선출원에서 소개한 전체연관심플코드, 부분연관심플코드, 자음연관심플코드, 음절기준이니셜코드, 단어기준이니셜코드 등의 심플코드 생성규칙을 활용할 수 있다. 예를 들어, 사용자가 "dance with the wolf"에 대한 심플코드를 생성함에 있어서, 단어기준 이니셜코드를 심플코드로 사용하고자 한다면, 심플코드 생성모드에서 "dance with the wolf" 를 입력하고, 그에 대한 단어기준 이니셜코드인 "3983"를 입력하여야 한다. 마찬가지로 "증권정보"에 대한 심플코드를 생성함에 있어서, 음절기준이니셜코드를 심플코드로 사용하고자 한다면, 심플코드 생성모드에서 "증권정보" 를 입력하고, "9196"을 입력하여야 한다.

그러나 만약 사용자가 특정 어구에 대하여 심플코드를 지정함에 있어서, 사용자가 원하는 심플코드의 유형을 시스템에 기억시키고 심플코드를 생성하고자 하는 어구를 입력함으로써, 굳이 특정 어구에 대한 코드를 입력하지 않고도 심플코드가 자동적으로 지정되도록 할 수 있다. 위의 예에서 음절기준 이니셜코드를 사용한다는 것을 시스템에서 설정하고, "증권정보"만 입력함으로써 "증권정보"에 대한 심플코드가 자동으로 "9196"으로 지정되도록 할 수 있는 것이다. 마찬가지로 음절기준 이니셜코드를 사용한다는 것을 시스템에서 설정하고, 先生(xiansheng)을 입력함으로써 先生(xiansheng)에 대한 심플코드가 자동으로 “97” 로 지정되도록 할 수 있는 것이다.

영어의 경우는 심플코드로 사용하고자 하는 알파벳을 부각하여 표시함에 있어서 대문자를 사용할 수 있다고 하였다. 따라서 심플코드 지정방식을 대문자를 심플코드로 사용하는 것

으로 미리 지정하고, "DaTe ToNignt" 입력시 자동으로 대문자 "DTTN" 에 대응되는 "3886"이 심플코드로 지정되도록 할 수 있고 혹은 "ToNignt ShoW" 입력시 자동으로 대문자 "TNSW"에 대응되는 "8679"가 심플코드로 지정되도록 할 수 있다.

<3> 6.7 심플코드를 활용한 어구의 단축입력방법 및 병행입력방법

<4> 심플코드(이니셜코드는 심플코드의 특별한 경우이므로 특별히 따로 언급하지 않는 한 심플코드에 포함되는 것으로 함)를 입력시 시스템(클라이언트측 시스템 혹은 서버측 시스템)에서 그 심플코드에 대응되는 어구로 인식하여 처리하는 것이 가능하다. 따라서 특정 심플코드에 대응되는 어구를 인식하여 사용자에게 다시 보여주면 이를 단어의 입력에도 활용할 수 있음은 당연하다.

<5> 실제로 외국의 알파벳 입력방법에서는 단어별로 "전체연관 심플코드"를 부여한 색인을 단말기(클라이언트측 시스템)에 저장하고 있으면서, 사용자가 코드 입력시 그에 대응되는 단어를 단어별 우선순위에 따라 보여주고 타겟단어를 확정할 수 있도록 함으로써 문자입력시스템을 구현하고 있다. 이에 대해서는 http://www.tegic.com 과 http://www.zicorp.com 의 두 인터넷 사이트를 참고할 수 있다. 이러한 방법을 편의상 본 발명에서는 "전체연관 단축입력방법" 혹은 "외국의 방법"이라고 하기로 한다. 혹은 이러한 접근방법을 사용하는 대표적인 입력시스템이 Tegic 사의 “T9” 이므로 “T9유사 입력방법(T9LIM : T9 Like Input Method)” 이라 부르기로 한다. 도5-2는 Tegic사의 T9 시스템에서의 입력사례를 보여준다. 도5-2에서와 같이 사용자가 meet 를 입력하기 위하여 “622 …” 을 입력하면 처음에는 시스템이 사용자에게 “off” 를 제공하나, “6228” 이 입력되면, “meet” 가 표시된다.

[96] Tegic社와 Zi社의 방법을 출원인이 제시하였던 키패드에서의 알파벳 입력방법과 비교하면, 출원인의 알파벳 입력방법은 알파벳 단위로 유니크한 코드를 부여하고, 풀코드에 의하여 타겟알파벳 혹은 타겟어구를 입력할 수 있도록 한 것이고, 위에서 언급한 외국의 방법은 단어단위로 전체연관 심플코드를 부여하고, 그 심플코드에 의하여 타겟단어를 입력할 수 있도록 한 것이다.

[97] 외국의 방법의 단점은 단어단위로 코드를 부여함으로써 미리 정의된 단어만을 입력할 수 있는 점, 다른 단어이면서 같은 코드를 가지는 경우 자주 사용되지 않는 단어의 입력을 위하여 토글버튼 혹은 이동버튼을 이용하여 타겟단어를 선택하고 확정함으로써 입력이 용이하지 않다는 점, 입력시 타겟단어가 아닌 단어가 일시적으로 출현할 수 있는 점, 시스템의 저장용량을 많이 차지하고 구현에 많은 비용이 소요된다는 점 등이 있다.

[98] 여기서 상용어구(상용단어 혹은 상용구절을 모두 포함하는 개념)에 대하여 심플코드(일부연관 혹은 전체연관)를 부여하고, 심플코드를 이용하여 타겟어구를 입력할 수 있음을 지적한다. 상용어구에 대한 심플코드는 시스템에서 기 정의되어 사용자에게 제공될 수도 있고 사용자가 임으로 지정할 수 있어야 함은 물론이다. 혹은 시스템에서 기 정의한 심플코드를 사용자가 임의로 수정할 수 있어야 한다. 사용자가 임으로 지정할 수 있는 것은 특정 상용어구에 대한 심플코드값을 사용자가 쉽게 알 수 있는 장점이 있다.

[99] 본 발명에서 심플코드(부분연관 심플코드, 전체연관 심플코드, 이니셜코드 모두 포함)를 이용하여 타겟어구를 입력하는 방법을 편의상 "단축입력방법"이라고 하며, 풀코드에 의하여 타겟알파벳을 입력하는 방법을 편의상 단축입력방법에 대비하여 "풀입력방법"이라고 한다. 또한 다음에서 설명하는 바와 같이 단축입력방법과 풀입력방법을 병행하여 적용할 수 있으며, 이

렇게 단축입력방법과 풀입력방법을 병행하여 적용하는 방법을 편의상 "단축/풀 병행입력방법" 혹은 간단히 "병행입력방법" 이라고 부르기로 한다.

10> 알파벳 단위로 유니크한 코드를 부여하고 그 코드를 입력하여 타겟알파벳을 입력할 수 있도록 하는 풀입력방법 중에서 특히 반복선택방법을 사용함으로써 발생하는 모호성을 편의상 "1차 모호성" 혹은 "알파벳모호성"이라 부르기로 한다. 반면에 외국의 방법에서처럼 모든 단어에 대하여 코드를 부여하고, 이 코드를 통하여 타겟단어를 입력할 수 있도록 하는 방식에서, 같은 코드에 대하여 서로 다른 여러 단어가 존재하는 모호성이 있는데, 이를 편의상 "2차 모호성" 혹은 "어구모호성"이라 부르기로 한다. 본 발명에서 단순히 모호성이라고 하면 1차모호성을 의미한다.

01> 특정 입력값에 대하여 1차적으로 심플코드로 해석하고(즉 1차적으로 단축입력방법을 적용 혹은 기본입력모드로 단축입력모드 적용), 입력값에 대응되는 심플코드가 없으면, 2차적으로 풀코드로 인식(즉 2차적으로 풀입력방법을 적용)하는 등의 시나리오 구성이 가능하며, 반대로 입력값에 대하여 1차적으로 풀코드를 이루는지 검사하고(즉 1차적으로 풀입력방법을 적용 혹은 기본입력모드로 풀입력모드 적용), 풀코드를 형성하지 못하면 심플코드로 인식(즉 2차적으로 단축입력방법을 적용)하는 등의 시나리오가 가능하다. 입력값에 대하여 1차적으로 심플코드로 해석하는 것은 기본입력모드로 "단축입력모드"를 적용하는 것으로 볼 수 있으며, 반대로 1차적으로 풀코드로 해석하는 것은 기본입력모드로 "풀입력모드"를 적용하는 것으로 볼 수 있다. 문자입력에 있어서 상용어구를 위주로 사용하는 사용자는 1차적으로 단축입력방법이 적용(즉 기본입력모드로 단축입력모드 사용)되도록 하는 것이 바람직하고, 그렇지 않은 사용자는

1차적으로 풀입력방법이 적용(즉 기본입력모드로 풀입력모드 사용)되도록 하는 것이 바람직할 것이다.

2> 여기서 기본입력모드를 풀입력모드로 설정한 경우, 입력값에 대하여 1차적으로 풀코드로 간주하여 해석하므로, 심플코드가 입력된 경우에도 그 입력값이 풀코드를 형성할 수 있으며, 사용자가 원하는 타겟단어가 아닌 다른 단어가 입력될 수 있다. 예를 들어, 도4-2에서 풀입력방법으로 기준반복선택방법을 적용할 경우, "옥수수" 라는 단어의 심플코드를 "877"로 하고(음절기준 이니셜코드 사용), 이를 입력시 1차적으로 풀코드로 간주하므로 이를 "여"로 인식할 수 있는 것이다. 이는 "옥수수", "옥수수를", "무진장", "와르르", "우수수", "와장창", "우당탕" 등과 같이 2, 3번째 글자의 초성이 동일한 버튼에 대응되는 경우에 있게된다. 반대로 기본입력모드로 단축입력모드로 설정한 경우, 입력값에 대하여 1차적으로 심플코드로 간주하여 해석하므로, 풀코드가 입력된 경우에도 타겟단어가 아닌 다른 단어로 인식될 수 있다. 즉 이는 심플코드와 풀코드간의 모호성인데, 이를 편의상 "3차모호성"이라고 부르기로 한다.

03> 이러한 3차모호성의 경우도, 기존의 방법과 유사하게 토글방법 혹은 리스트에서 선택하는 방법으로 극복할 수 있다. 혹은 다른 방법이 있을 수 있는데, 이러한 모호성이 발생할 수 있는 입력값을 입력하기 전에 단어단위로 풀입력모드에서 단축입력모드로 혹은 단축입력모드에서 풀입력모드로 전환할 수 있도록 하는 것이다. 이는 출원인의 선출원에서 제시한 바 "아/아"컨트롤을 두어 단어단위의 히라가나/가타가나 전환 컨트롤을 두고 이를 선택함으로써, 히라가나모드에서 가타가나 한 단어를 입력하거나 혹은 가타가나 모드에서 히라가나 한 단어를 입력할 수 있도록 하는 것과 유사한 것이다. 예를 들어, 기본입력모드가 풀입력모드인 경우, "단축/풀" 컨트롤을 두고 이를 선택한 후 입력되는 입력값에 대하여, 시스템은 이를 처음부터 심플코드로 인식하고 색인을 참조하여 입력값에 해당하는 타겟단어를 사용자에게 제공할 수 있

는 것이다. 단축입력모드를 기본입력모드로 사용하는 경우에도 마찬가지로 "단축/풀" 컨트롤 선택후 입력되는 입력값을 처음부터 풀코드로 인식하여 처리할 수 있다. "단축/풀" 컨트롤은 마찬가지로 타겟단어에 대하여 선입력될 수도 있고 후입력될 수도 있으나, 이 경우는 선입력하도록 하는 것이 편리할 것이다.

4> 병행입력방법 적용시의 입력값이 풀코드인지 심플코드인지의 판단은 앞에서 1차모호성을 제거하기 위하여 색인을 참조하는 것처럼 단어단위로 이루어질 수도 있고, 다음과 같이 입력중간에 입력값이 풀코드인지 심플코드인지 판단될 수도 있다.

15> 풀입력방법을 기본모드로 하고 병행입력방법을 적용시, 하나하나의 코드입력시마다, 입력값이 풀코드를 형성하는지 조사하여 풀코드를 형성하지 못하는 것으로 판단되는 시점에서 입력값을 심플코드로 판단하여 심플코드 색인을 참조하여 그에 대응되는 어구를 사용자에게 보여줌으로써 병행입력방법의 효율을 배가시킬 수 있다. 단축입력방법을 기본모드로 병행입력방법을 적용할 경우도 마찬가지로 하나하나의 코드입력시마다 입력값과 색인을 비교하여 색인에 있는 입력값과 일치하는 단어가 없는 것을 확인하는 순간 이를 기 설정된 풀입력방법의 풀코드로 판단하여 처리할 수 있다. 이는 입력중간에 발생하는 3차 모호성을 각각의 풀입력방법에서의 규칙을 적용하여 입력초기에 제거할 수 있음을 의미한다. 이는 풀입력방법으로 선출원에서 제시하지 않은 알파벳입력방법을 사용할 경우도 마찬가지로 적용될 수 있다. 선출원에서 제시하였던 풀입력방법(기준반복선택방법, 부분전체선택방법)을 기준으로 예를 들면 다음과 같다.

106> 한국어의 경우 예를 들면, 풀입력방법으로 도4-2을 기준으로 기준반복선택방법을 적용하였을 경우, 풀코드에 의한 모든 음절의 2번째 입력값과 3번째 입력값은

항상 같은 값을 가져야 하므로 이러한 조건에 위배되는 경우 입력값을 심플코드로 판단하여 처리할 수 있다. 쌍자음을 기본자음의 조합으로 처리하는 것을 허용하는 경우는 그에 따른 판단기준을 적용할 수 있다. 이는 기준반복방법을 적용하는 모든 경우에 적용가능하다.

07> 또한 모든 언어에 있어서, 풀입력방법으로 부분전체선택방법을 적용시, 하나의 알파벳에 2개의 입력값이 대응되며, 그 2개의 입력값 중 1번째 입력값에 대하여 2번째 입력값은 제한되어 있다. 예를 들어, 영어의 경우 도1-1에서와 같이 좌우직선조합만을 사용하는 경우, 숫자를 부분전체선택방법으로 선택하지 않도록 하면, 첫번째 행의 버튼([1], [2], [3])이 하나의 알파벳에 대응되는 1번째 입력값으로 사용되었을 경우 2번째 입력값으로 올 수 있는 값은 역시 1번째 행의 버튼([1], [2], [3])인 것이다. 마찬가지로 [2]+[1] 이 입력되고 나서 다시 두번째 행의 버튼([4], [5], [6])이 입력되면, 풀코드를 형성하기 위하여 다음에 올 수 있느 값은 역시 두번째 행의 버튼([4], [5], [6]) 중의 하나인 것이다. 입력값이 이를 위배하는 순간 시스템에서는 입력값을 심플코드로 간주하고, 그 심플코드에 대응되는 단어를 사용자에게 우선순위에 따라 추천하여 줄 수 있는 것이다. 예를 들어 도5-3에서와 같이 부분전체선택방법을 풀입력방법으로 하고, 풀입력방법을 기본입력모드로 하는 병행입력방법에서, 사용자가 “help” 의 심플코드를 4357 입력시, 2번째 입력값 3이 입력되는 순간 입력값이 풀코드를 이루지 못하는 것을 시스템이 인식하고, 입력값을 심플코드로 간주하여 처리할 수 있는 것이다.

208> 만약 도4-2에 대하여 부분전체선택방법을 적용한다면, 풀코드를 형성하기 위하여 최초 [1] 버튼의 입력에 대하여 올 수 있는 다음 버튼은 [1] 또는 [2] 버튼 밖에 없다. 이를 위배하였을 경우 입력값이 풀코드가 아닌 심플코드인 것으로 판단하여 색인을 참조하여 입력값에 대응되는 타겟단어를 우선순위에 따라 사용자에게 추천하여 줄 수 있다. 도1-3에서와 같이 P, Q, R, S 의 4개 알파벳이 [7] 버튼에 배정되는 경우, 선출원에서 지적한 바와 같이 4개의 알파

벳 중 하나의 알파벳을 상하인접조합을 이루는 격자에 배치할 수 있으며, 이 경우에도 하나의 알파벳을 위한 풀코드를 이루기 위하여 [7] 버튼이 입력되면 다음에 올 수 있는 버튼은 3번째 행의 버튼([7], [8], [9]) 혹은 상하인접조합의 버튼([4] 버튼)만이 올 수 있으므로, 이에 위배되는 경우 입력값을 심플코드로 처리할 수 있다. 이는 선출원에서 제시한 부분전체선택방법을 적용하는 경우에 모든 언어에 대하여 적용가능하다.

09> 예를 들어 도1-1을 기준으로한 심플코드 색인에 北京(Beijing)의 심플코드가 음절기준 이니셜코드로써 "25" 이 저장되어 있는 상태에서, 풀입력방법으로 부분전체선택방법을 사용하고, 풀입력모드를 기본입력모드로 하여, 사용자가 "25" 을 입력하였을 경우, 시스템은 처음(단어시작시점 부터) [2] 입력 후 [5]가 입력되는 순간 입력값 "25" 가 풀코드를 이룰 수 없는 것([2] 다음에 올 수 있는 좌우직선조합의 버튼은 [1], [2], [3] 중의 하나이므로)을 인식하고, 심플코드 색인을 참조하여 입력값 "25" 에 대응되는 "北京" 을 사용자에게 제공할 수 있는 것이다. 중국어의 경우 특별히 예시한 심플코드 "25" 에 대응하는 어구로 한자(漢字, 즉 北京)를 사용자에게 제공할 수 있다(한자 "北京" 이 타겟단어이므로). 중국어가 아닌 소리문자를 사용하는 언어에 있어서는 심플코드 "25" 에 대응하는 "Beijing" 를 시스템이 사용자에게 제공하게 될 것이다. 심플코드 "25" 에 대응하는 어구가 여러 개 있다면, 목록의 형태로 제공되거나 특정버튼을 반복하여 누름으로써(토글방식에 의하여) 사용자가 원하는 어구를 선택하도록 할 수 있다.. 도5-4를 참고한다.

210> 결국 도5-4와 같이 병행입력방법을 적용하는 것은 "北京(Beijing)" 과 같이 자주 사용하는 단어에 대한 심플코드를 색인에 등록하여 두고, 일반적인 풀입력방법과 단축입력을 모드전환없이 병행하되, 심플코드로 등록된 단어를 입력시 적은 타수로 입력할 수 있도록 하는 것이다.

1> 도4-5의 경우는 버튼당 기본자음이 하나씩만 배정되어 있으므로, 한국어에서 범용하게 사용할 수 있는 음절기준 이니셜코드를 심플코드로 사용할 경우, 단축입력방법과 풀입력방법을 병행하여 적용시에도 3차 모호성 없이 입력이 가능한 좋은 특성이 있다. 즉 병행입력방법 적용시 사용자가 음절기준 이니셜코드를 입력하면, 2번째 입력(경음, 격음을 컨트롤처리방법에 의하여 입력할 경우)부터 입력값이 풀코드를 이룰 수 없게 되므로, 심플코드의 색인을 검색하여 우선순위에 따른 적절한 단어를 사용자에게 추천하여 줄 수 있는 것이다. 풀코드 입력의 경우도 같다.

12> 위에서 예를 든 바와 같이, 병행입력방법 적용시 입력중간에 입력값이 심플코드인지 풀코드인지 판단할 수 있는 것이 본 발명의 "핵심"중의 하나이다. 이는 출원인이 제시한 풀입력방법을 사용하는 경우 뿐만이 아니라 다른 풀입력방법을 사용하는 경우에도 적용된다. 한가지 예를 예를 들면, 각 알파벳이 배정된 제1버튼과 제1버튼에서의 알파벳의 순위에 해당하는 제2버튼을 조합하여 알파벳을 입력하는 방법이 있었다. 즉, 도1-1에서 P = [7]+[1] 로 입력하는 것이었다. 이 경우 제2버튼은 도1-1에서 [1] , [2], [3] 버튼중의 하나가 되므로, 입력값이 이 규칙을 위배하는 순간, 시스템은 입력값을 심플코드로 간주하여 처리할 수 있는 것이다. 특히 출원인이 제시한 풀입력방법은 위의 예시에서와 같이 병행입력방법 적용시 입력값이 풀코드를 이루는지를 입력중간에도 쉽게 알 수 있는 좋은 특성이 있다.

13> 또한 출원인의 선출원에서 언급한 바와 같이 심플코드 혹은 풀코드를 해석하는 것은 클라이언트측에서 일어날 수도 있고, 서버측에서 일어날 수도 있다. 그리고 선출원에서 1차모호성(알파벳 모호성)을 극복하기 위한 색인을 룩업하는데 있어서, 1차적으로 클라이언트측의 색

인을 참조하고 2차적으로 서버측 색인을 참조하는 시나리오를 심플코드 혹은 풀코드를 해석하는 데 있어서도 적용할 수 있다. 반대로 1차적으로 서버측 색인을 참조하고, 2차적으로 클라이언트측 색인을 참조하는 것도 가능하다. 또한 더 나아가 입력값을 먼저 심플코드로 해석하되, 1차적으로 클라이언트 측의 색인을 참조하고, 2차적으로 서버측 색인을 참조하며, 입력값에 대한 심플코드를 검색하지 못하면, 다시 풀코드로 해석하되, 1차적으로 클라이언트 측에서 해석하고, 클라이언트측에 이러한 해석기능이 없으면, 2차적으로 서버측에서 해석하는 것이 가능하다. 다른 예로 입력값을 1차적으로 심플코드로 해석하되, 클라이언트측의 색인과 서버측의 색인을 모두 참조하여, 사용자에게 제공하고 타겟단어를 사용자로 하여금 선택하도록 하는 것도 가능하다. 이와 같이 해석방법(심플코드, 풀코드)과 해석장소(클라이언트측, 서버측)를 적용함에 있어서 이와 유사한 변형이 가능하다. 즉 정리하면, 해석방법(심플코드, 풀코드)과 해석장소(클라이언트측, 서버측)의 적용에 있어서 다양한 조합이 가능하다는 것이다. 즉 입력값에 대하여 도5-5에서 (A)-(B)-(C)-(D)의 순서로 혹은 (A)-(C)-(B)-(D)의 순서뿐만 아니라 (A), (B), (C), (D)의 모든 다양한 조합에 의한 순서로 적용이 가능하다는 것을 의미한다.

<114> 이렇게 단축입력방법과 풀입력방법을 병행하여 적용하는 장점은 다음과 같다. 우선 풀입력방법을 사용할 수 있으므로 사용자는 기 정의된 단어 이외의 사전에 존재하지 않는 단어라도 모든 단어를 입력할 수 있으며, 상용어구에 대하여 단축입력방법의 사용을 위한 심플코드를 편의대로(부분연관/전체연관) 지정할 수 있고, 부분연관 심플코드를 지정하여 둠으로써 입력타수를 대폭 줄일 수 있다. 그리고 단어에 대해서 뿐만 아니라 구절에 대해서도 단어기준 이니셜코드를 부여할 수 있는 장점이 있다. 그러나 외국의 방법에서는 모든 단어의 입력에 단어

단위의 색인을 이용한 방법을 사용하므로, 서로 다른 단어에 대하여 같은 코드가 부여되는 경우를 최소화 하기 위하여 전체연관 심플코드를 사용할 수 밖에 없었다.

5> 또한 시스템이 특정 상용어구와 그 상용어구에 대한 코드값을 가지고 있는 "색인"을 가지고 있어야 하는데, 모든 단어에 대한 "색인"을 가지고 있는 외국의 방법에서 보다 훨씬 더 적은 용량의 저장장소를 필요로 한다. 그리고 이 "색인"은 모호성을 없애기 위하여 모호성이 발생할 수 있는 단어에 대하여 올바른 단어에 대한 "색인" 혹은 그른 단어에 대한 "색인"을 시스템에서 가지고 있어야 하는데, 이 모호성을 없애기 위하여 준비하는 색인과 공동으로 사용될 수 있다.

16> 모든 언어에 있어서 특정 단어의 음가를 가지고 있는 것은 자음이므로 자음을 추출하여 약어를 만드는 기법은 이미 널리 사용되고 있었다. 영어의 경우 예를 들면, 군대용어중 " captain" 에서 음가를 가지고 있는 자음 "CPT"를 추출하여 약어로 사용하고, "private" 에서 " PVT"를 약어로 사용하고, "sergent"에서 "SGT"를 약어로 사용하고, "staff sergent"에서 "SSG" 를 약어로 사용하고, "sergent first class"에서 SFC를 약어로 사용하는 등등이다. 물론 예로 든 "captain" 과 "private" 는 각각 2음절로 이루어져 있지만, 여기서 약어로 추출된 자음은 각 음절을 대표하는 자음으로 볼 수 있다. 따라서 "captain" 의 경우는 "CPT"와 연관된 "278" 을 심플코드로 지정할 수 있는 것이다.

17> 이렇게 음절을 기준으로 임의로 상용어구에 대하여 부분연관 심플코드를 지정하여 단축입력방법을 적용할 수 있는 것은 입력노력을 감소시키는 이외에도 큰 의미가 있다. 음성학적으로 음절은 "심리적 실체"라고 규정하고 있다. 또한 음절에서 음가를 가지고 있는 것은 자음인 것이다. 예로 든 captain 에서 모음

"AAI"만을 추출하였을 경우 captain 을 유추해 내는 것은 거의 불가능하다. 그러나 자음 "CPTN" 혹은 "CPT"를 추출하면 쉽게 captain 을 유추해 낼 수 있다. 영어의 어떤 문장에서 각 단어별로 모음을 제거하고 자음만을 나열해 놓아도 보통의 경우 원래의 문장을 유추해 낼 수 있다고 한다. 즉 음절을 이루는 주체인 각 자음과 연관하여 부분연관 심플코드를 사용하는 것은 사용자가 자연스럽게 단축입력방법에 적응할 수 있음을 의미하고, 사용자에게 편리를 줄 수 있음을 의미한다.

8> 특히 영어권에서는 약어를 사용하는 것이 일반화되어 있고, 상장회사명과 같은 경우는 일정한 개수로 이루어진 약어가 지정되어 있으므로, 이 약어에 기초하여 심플코드를 활용할 수 있다.

9> 특정 어구에 대하여 사용자가 편의대로 심플코드(부분연관, 전체연관)를 지정할 수 있도록 하는 것은 상용어구에 대한 코드값을 기억하기 용이하도록 하는 장점이 있다. 더 나아가 만약 어떤 사용자가 극히 일부의 상용어구만을 필요로하는 사용자라면(예를 들어 10개 이하의 상용어구만을 사용), 상용어구의 알파벳에 연관된 코드를 굳이 부여하지 않고, 단지 각 상용어구에 1, 2, 3, . . . 등과 같이 심플코드를 부여할 수도 있다.

20> 6.8 심플코드/대응어구의 그룹핑 및 검색범위 지정

21> 다수의 어구에 대하여 심플코드를 지정하면, 많은 중복성이 발생할 가능성이 있는데, 심플코드에 대응되는 어구들을 그룹핑을 하여, 특정 어구그룹에 대해서만 심플코드를 검색하도록 하여 심플코드간의 모호성(2차모호성)을 감소시킬 수 있다. 하나의 어구가 반드시 하나의 그룹에만 속해있어야 하는 것은 아니고 여러 그룹에 속해 있을 수도 있다.

[222] 예를 들어, 심플네이밍한 어구들을 상장회사명, 도시명, 상용어구, … 등의 그룹으로 그룹핑하고 다시 상용어구 그룹을 사회분야, 정치분야, ... 등의 세부그룹으로 하여 트리형태로 그룹핑할 수 있다. 본 예에서는 2단계의 트리형태의 그룹핑을 제시하였으나, 3단계, 4단계 혹은 그 이상 단계로 트리형태의 그룹핑이 가능하다. 도5-6을 참고한다. 사용자가(혹은 시스템이) 심플코드의 검색범위를 상장회사명 그룹으로 한정하면, 특정 심플코드 입력시 시스템은 상장회사명 그룹의 범주에서만 입력된 심플코드에 대응되는 네이밍된 어구를 검색하므로 2차모호성을 감소시킬 수 있다. 마찬가지로 검색범위를 상용어구로 하면 상용어구 그룹이하의 모든 세부그룹을 포함하여 검색범위로 되도록 할 수 있다. 만약 상용어구 그룹중 사회분야 그룹을 검색범위로 지정하면, 사회분야를 포함하는 그 이하(트리형태의 그룹구조에서)의 그룹만이 검색범위로 될 것이다.

[223] 도5-6에서의 트리형태의 그룹은 윈도우의 탐색기에서 폴더처럼 생각하면 된다. 검색범위는 시스템에 의하여 자동으로 재지정 될 수도 있다. 예를 들어, 도시명의 서브그룹으로 행정구역의 계층구조가 저장되어 있을 때, 사용자가 특정 도시를 선택하고 나면, 그 다음 입력값에 대한 검색범위는 당연히 특정도시의 하위 행정구역 단위의 명칭이 될 것이다. 예를 들어, 사용자가 도시중 "서울" 을 선택하고 나면, 다음 선택범위는 서울시에 있는 "구" 가 되며, 다시 사용자가 "양천구" 를 선택하고 나면, 다음 선택범위는 양천구내에 있는 "동" 이 되는 것이다.

[224] 6.9 중계서버의 활용

[225] 심플코드의 해석은 클라이언트측에서 이루어질 수도 있고, 서버측에서 이루어질 수도 있는데, 이러한 심플코드(경우에 따라 풀코드 포함)의 해독을 전담하여 심플코드에 대응되는 어

구를 클라이언트측 혹은 다른 서버측에 제공하는 중계서버를 둘 수 있다. 도6-1를 참고할 수 있다. 또한 도6-1에서 클라이언트측에서 1차적으로 심플코드를 해독하고 입력된 심플코드에 대응하는 어구를 해석하지 못하면, 2차적으로 중계서버에서 입력된 심플코드에 대응하는 어구를 해석하고 해석실패시, 다시 3차적으로 각각의 서버에서 입력된 심플코드에 대응되는 어구를 해석하는 것이 가능하다. 3차 심플코드 해독 서버(편의상 "3차서버"라 함)는 입력된 심플코드 혹은 심플코드에 대응되는 어구를 사용하는 어플리케이션을 탑재한 서버가 된다.

26> 중계서버를 두면, 3차서버측에서는 심플코드가 아닌 어구를 필요로 하는 경우에도, 사용자가 심플코드를 입력하고, 3차서버측에서 심플코드와 심플코드에 대응되는 어구를 저장하고 있지 않아도, 중계서버가 사용자로부터 입력된 심플코드를 중계서버가 해석하여 심플코드에 대응되는 어구를 3차서버측에 전달해 주는 것이 가능하다.

27> 심플코드의 코드값 하나 하나가 입력될 때마다, 심플코드에 대응되는 어구가 저장된 색인을 검색하여 클라이언트측 혹은 각 서버측에 피드백을 줄 수도 있고 단어단위로 피드백을 줄 수도 있다.

28> 6.10 단어단위의 구분

29> 본 발명에서 단어단위라 함은 단어시작에서부터 단어종료까지를 의미한다. 이는 단어와 단어의 구분을 줄 수 있는 모든 요소(단어시작, 공백, 모드전환, 엔터, …)의 조합을 통하여 식별될 수 있다. 예를 들어 공백 ~ 공백, 공백 ~ 모드전환 등은 단어시작 ~ 단어종료 등을 통하여 단어가 입력되었음을 식별할 수 있다. 단어단위로 피드백을 주는 것은 현재 네트웍 환경을 지원하는 언어를 통하여 충분히 구현될 수 있다.

30> 6.11 심플코드/대응어구의 다운로드

31> 더 나아가 서버측에 있는 심플코드와 심플코드에 대응하는 어구를 사용자가 직접 입력하여 저장시키지 않아도, 클라이언트 측으로 다운받을 수 있도록 할 수 있다.

32> 다운받는 단위는 하나 하나의 어구 단위로 다운받을 수도 있고, 위에서 예를 든 어구의 그룹 단위(트리구조의 그룹)로 다운받을 수도 있다. 상위의 그룹을 선택시 하위의 그룹까지 다운받을 수 있다. 서버측 어구그룹의 트리구조를 그대로 유지하면서 다운받을 수도 있고, 해당그룹과 하위그룹에 속한 어구들을 클라이언트측에 사용자가 지정하는 하나의 그룹으로 하여 다운받을 수도 있다. 심플코드의 해독을 주기능으로 하는 중계서버가 있다면, 중계서버에서 이러한 기능을 담당할 수도 있다.

33> 7. 기호의 입력

34> 선출원에서 제시하였던 바, "기준격자에 가까운 순서"로 자국어 알파벳을 배치하고, 다음으로 숫자 그리고 영어알파벳을 배치할 수 있음을 설명하였고, 기준반복선택방법 적용시 역시 기준격자에 가까운 순서로 이를 자국어 알파벳, 숫자가 선택되도록 할 수 있음을 설명하였다. 마찬가지로 후속컨트롤처리방법을 적용함에 있어서, 특정버튼에 속한 자국어 알파벳 뿐만아니라, 숫자, 영어알파벳도 후속컨트롤처리방법을 통하여 입력할 수 있다.

<5> 본 발명에서는 한 걸음 더 나아가 선출원에서 키패드상에 표시하여 입력할 수 있도록 하였던, 각종 기호 등을 키패드상에 표시하지 않고도 효율적으로 입력할 수 있는 방법(즉, 은닉형 컨트롤처리방법에 의하여)을 제시한다.

<6> 즉 선출원에서 컨트롤들을 배치하였던 격자중 적절한 격자에 "기호컨트롤"을 두고, 기호컨트롤과 기호를 의미할 수 있는 버튼(컨트롤 버튼이 아닌 다른 버튼)을 조합하여 기호를 입력하는 것이다. 여기서 기호를 의미할 수 있는 버튼의 예를 들면, 마침표의 경우 마침표에서 "ㅁ"을 쉽게 연상할 수 있으므로 [5] 버튼이 된다.

<37> 예를 들어 선출원의 발명 내용 중 한국어의 실시예(도4-2)에서 기호컨트롤을 [*] 버튼을 2번 연속누름으로 선택할 수 있는 격자의 위치에 둘 수 있다. 즉 ㄱ(대표알파벳), ㅋ(2nd), 기호(3rd), ... 와 같이 대표알파벳과 후속알파벳의 관계를 두는 것이다. 컨트롤 선입력의 경우, "마침표"를 입력시, ㅁ = {기호}+ㅁ = [*]+[*]+[5] 이 된다. 컨트롤 후 입력시, ㅁ = ㅁ+{기호} = [5]+[*]+[*] 이 된다. 만약, 기호컨트롤을 [*] 버튼을 3번 눌러서 선택할 수 있는 격자의 위치에 두었을 경우는 예에서 [*] 버튼의 입력을 하나씩 더하면 된다.

<38> 기호컨트롤을 하나만 두었을 경우, 10개의 각 숫자버튼마다 기호의 의미를 한가지씩 부여하더라도, 10개 정도의 기호만을 입력할 수 있다. 각 버튼에 기호의 의미를 부여한 예를 들면 다음과 같다.

<39> [1] 버튼 : ? 의 의미 부여 ("ㄱ" 이 배치되어 있는데 ? 와 모양이 유사)

<40> [2] 버튼 : ! 의 의미 ("ㄴ"이 배치되어 있는데 "느낌표"의 첫글자 동일)

<41> [3] 버튼 : $ 의 의미 ("ㄷ"이 배치되어 있는데 "달러"의 첫글자 동일)

<42> [4] 버튼 : . . .

3> [5] 버튼 : . 의 의미 ("ㅁ"이 배치되어 있는데 "마침표"의 첫글자 동일)

4> [6] 버튼 : * 의 의미 ("ㅂ"이 배치되어 있는데 "별표"의 첫글자 동일)

5> [7] 버튼 : , 의 의미 ("ㅅ"이 배치되어 있는데 "쉼표"의 첫글자 동일)

6> [8] 버튼 : " 의 의미 ("ㅇ"이 배치되어 있는데 "인용부호"의 첫글자 동일)

7> [9] 버튼 : ~ 의 의미 (모음 "ㅡ"가 배치되어 있는데 ~ 와 모양이 유사)

8> [0] 버튼 : @ 의 의미 (숫자 0 이 배치되어 있는데 @와 모양 유사)

9> 위에서와 같이 각종의 기호를 연상시킬 수 있는 버튼과 기호컨트롤을 조합함으로써 기호를 입력할 수 있다. 각 버튼에 기호의 의미를 부여하는 것은 예에서와 같이 기호의 자국어 명칭/모양과 배치된 알파벳의 연관관계, 혹은 영어 명칭/모양과 배치된 영어 알파벳과의 연관관계, 배치된 숫자 명칭/모양과의 연관관계 등을 고려하여 기호의 의미를 부여할 수 있다. 이는 개인의 취향에 따라 다를 수 있으므로 반드시 위와 같이 되어야 하는 것은 아니고, 개개인으로 하여금 각 버튼별로 기호의 의미를 (재)설정하도록 할 수도 있다.

50> 이를 통하여 예시에서와 같이 빈번히 쓰이는 기호를 연상이 용이한 숫자버튼에 속한 후속알파벳인 것처럼 처리할 수 있다. 위의 예에서 물음표를 모양의 유사성을 고려하여 "ㄱ"이 배치된 [1] 버튼과 연관시킨 것은 물음표보다 더 빈번히 쓰이는 마침표가 "ㅁ"으로 시작하기 때문이다.

251> 마찬가지로 버튼별로 기호의 의미를 부여하는데 있어서, 영문명칭/모양 혹은 숫자명칭/모양을 이용할 수도 있다. 아래의 예는 자국어의 경우와 혼용하여 적용할 수도 있다.

[52] [1] 버튼 : ? 의 의미 (알파벳 "q" 가 배치되어 있는데 "Question mark"의 첫글자 동일)

[53] [2] 버튼 : , 의 의미 ("c"가 배치되어 있는데 "Comma"의 첫글자 동일)

[54] [3] 버튼 : . 의 의미 ("d"가 배치되어 있는데 "Dot"의 첫글자 동일)

[55] [4] 버튼 : ! 의 의미 ("i"가 배치되어 있는데 "!"와 모양 유사)

[56] [5] 버튼 : . . .

[57] [6] 버튼 : . . .

[58] [7] 버튼 : / 의 의미 ("s"가 배치되어 있는데 "Slash"의 첫글자 동일)

[59] [8] 버튼 : : 의 의미 (숫자 "8"이 배치되어 있는데 : 과 모양 유사)

[60] [9] 버튼 : ! 의 의미 ("x"가 배치되어 있는데 "eXclamation mark"와 발음 연관)

[61] [0] 버튼 : @ 의 의미 (숫자 0 이 배치되어 있는데 @와 모양 유사)

[62] 영어를 기준으로 기호의 의미를 부여하는 것은 영어가 혼용하여 표기된 키패드를 사용하는 비영어권에서도 유용하게 사용될 수 있다. 또한 colon 을 숫자 8 의 모양의 유사성을 두어 설정하는 것 역시 언어에 상관없이 범용하게 사용될 수 있는 장점이 있다. 마찬가지로 comma 역시 숫자 9 와의 모양의 유사성을 두어 [9] 버튼에 속한 후속알파벳으로 간주할 수도 있으나, 위의 예에서는 그러하지 아니하였다.

[63] 기호컨트롤은 적절한 버튼에 설정하면 된다. 영어의 경우 도1-1에서 [*] 버튼에 다른 용도의 컨트롤을 배치하지 않는다면, [*] 버튼 1타로 기호컨트롤을 선택하도록(즉 [*]버튼 기준격자의 위치에 기호컨트롤을 두는 것) 할 수도 있다. 그리고 첨자가 붙은 변형알파벳이 존

재하는 유럽 각국어의 경우는 [0] 버튼 혹은 [#] 버튼의 기준격자의 위치에 기호컨트롤을 둘 수 있다. 단 [0] 버튼에 기호컨트롤을 두는 경우는 위의 예에서 처럼 "@" 기호의 의미를 부여하지 않는 것이 좋다.

)4> 기호컨트롤을 [*]버튼에 두고 컨트롤후입력을 적용하면, 영어의 경우 도1-1에서 colon(:) 입력시, : = [8]+{기호} = [8]+[*] 과 같이 된다.

35> 위의 예에서 보듯이 각 버튼에 연관되는 기호의 의미를 하나씩 부여하고 기호컨트롤을 컨트롤 버튼의 어느 한 격자에만 두는 경우는 약 10개의 기호만을 입력할 수 있다. 그리고 각 버튼에 의미를 부여하는데 있어서도 영어의 예에서와 같이 "s" 가 배정된 버튼에 slash, semi-colon, period 등의 의미를 부여할 수도 있으나, 그 중 하나인 slash 의 의미만을 부여하였다. "d" 가 배정된 버튼에도 dot의 의미를 부여하였으므로, exclamation mark 의 경우 부득이 모양의 유사성을 고려하여 "i"가 배정된 버튼에 "!" 의 의미를 부여하였다.

)66> 따라서 기호컨트롤을 다수 두어(예를 들어 기호컨트롤1, 기호컨트롤2, . . .), 컨트롤을 선택할 수 있도록 함으로써, 더 많은 수의 기호를 컨트롤처리방법을 통하여 입력할 수 있다. 예를 들어, "d" 가 배정된 버튼에 dot 의 의미를 부여하고(혹은 "ㅁ"이 배정된 버튼에 "마침표"의 의미를 부여하고), 모양이 유사한 comma 를 dot의 후속알파벳인 것처럼 처리할 수 있다.

267> 격자음 또는 경자음이 존재하는 기본자음이 배정된 버튼에 연관시킨 기호 입력의 예를 들면 “dot(.) = 3****” 로 할 수 있다. “3*** = ㅋ” 로 되기 때문에 [*]버튼 4번 약속된 기호를 입력할 수 있다. “comma(,) = 3*****” 와 같이 dot(.)에 입력 후 [*]버튼을 추가로 한

번 더 눌러 입력이 가능한데, 이는 한국어 음절이 모음 "ㅡ"로 시작하지 않는 성질을 이용한 것이다. comma(,)의 입력은 체인형 후속컨트롤처리방법의 관점에서 이해할 수 있다.

68> 만약 격음을 컨트롤처리방법에 의하여 입력하지 않는 것으로 한다면(즉 격음컨트롤이 없는 것으로 간주한다면), [*] 버튼2타에 "기호컨트롤1"이 선택됨은 당연하다(건너뛰기 컨트롤처리). 마찬가지로 모양이 유사한 colon 과 semi-colon 도 같은 버튼에 배정된 후속알파벳들로써 간주하고, 컨트롤처리방법에 의하여 입력할 수 있다. 이외의 다른 기호들에 대해서도 이와 유사한 요령으로 적용할 수 있다.

69> 이렇게 "기호컨트롤1"과 "기호컨트롤2"로 2개의 기호컨트롤을 사용하는 경우도, 각각의 버튼에 기호의 의미를 부여하고 이를 기억하고 있어야 하며, 입력할 수 있는 기호의 갯수에 한계가 있게 된다. 따라서 dot와 comma를 그리고 colon과 semi-colon 을 그룹핑한 것과 같은 요령으로 기호를 그룹핑하고, 다수의 기호컨트롤을 둠으로써, 더 많은 기호를 입력할 수 있도록 할 수 있다.

70> 기호들을 그룹핑하는 방법 역시 사용자의 편의대로 할 수 있도록 하는 것이 바람직하다. 본 발명에서는 일반적으로 사용될 수 있는 기호 그룹핑의 예를 보인다. 먼저, 각종의 기호의 모양의 형태를 기준으로 dot 의 변형형태를 하나의 그룹으로 설정할 수 있다. 예를 들어, dot(.), comma(,), colon(:), semi-colon(;), quotation mark("), . . . , question mark(?), exclamation mark(!), . . . 와 같이 그룹핑하는 것이 가능하다. 이는 dot 형태, 즉 "0차원" 형태의 기호를 그룹으로 그룹핑 한 것이다. 여기서 ? 와 ! 는 0 차원 형태와 1차원 형태를 동시에 가지고 있으므로, 0차원 형태(dot 형태)의 그룹에 포함시킨 것이다. 그룹의 부속순위를 정함에 있어서는 선출원에서 언급한 바와 같이 사용빈도 등을 고려할 수 있으나, 역시 사용자로 하여금 설정하도록 할 수 있는 것이 좋을 것이다. 또한 후속알파벳으로 간주하는 기호의

갯수가 많을 때는 기호컨트롤을 후입력하도록 하는 것이 편리할 것이고, 디스플레이 창이 있는 단말기에서는 컨트롤버튼의 누름에 따라 기호가 변하는 것을 확인할 수 있다.

1> 이렇게 그룹핑된 dot 형태(0차원 형태)의 기호를 특정 버튼에 연관시키는 것 역시 사용자의 편의대로 설정할 수 있도록 할 수 있다. 예를 들어, 그룹을 대표할 수 있고 가장 빈번히 쓰이는 기호가 dot 라고 하면 dot 의 "d"를 포함하고 있는 [3] 버튼에 속한 후속알파벳으로 간주할 수 있다. 도1-1에 기호컨트롤을 추가한 도7-1의 예에서 기호컨트롤 버튼을 [*] 버튼으로 하고, 컨트롤 후입력을 적용하면, dot = [3]+[*], comma = [3]+[*]+[*], colon = [3]+[*]+[*]+[*], semi-colon = [3]+[*]+[*]+[*]+[*], . . . 과 같이 된다. 혹은 0차원 형태이므로 [0] 버튼에 속한 후속알파벳으로 간주한다면, [3] 버튼이 아닌 [0] 버튼과 결합하게 될 것이다. 아니면, dot 형태는 가장 기본적인 형태이므로 [1] 버튼에 속한 후속알파벳으로 간주할 수도 있다.

72> 다음으로 line 형태, 즉 "1차원" 형태의 기호를 그룹으로 그룹핑 할 수 있다. 예를 들어, slash(/), 모자기호(^), question mark(?), exclamation mark(!), 둥근괄호1((), 둥근괄호2()), 꺾쇠괄호1(<), 꺾쇠괄호2(>), 직각괄호1([), 직각괄호2(]), 물결기호(~), minus(-), 화살표1(<-), 화살표2(->), . . . 등과 같이 그룹핑할 수 있다. 역시 선출원에서 설명한 바와 같이 사용빈도 등을 고려하여 부속순위를 두면되며, 이를 특정버튼의 후속알파벳으로 연관시키는 것은 앞에서 설명한 바와 같이 적절한 버튼을 설정할 수 있다. 예를 들어 [1] 버튼에 속한 후속알파벳으로 간주하거나 혹은 알파벳 "l"이 배치된 [5] 버튼의 후속알파벳으로 간주하는 것이다.

3> 그 다음, line의 결합형태, 즉 "2차원" 형태의 기호를 그룹으로 그룹핑 할 수 있다. 예를 들어, at(@), ampersent(&), asterisk(*), sharp(#), dollar($), 원화기호(₩+=), 엔화기호(¥), . . . , heart1(♡), heart2(♥), 클로버1(♧), 하얀세모1(◁), 하얀세모2(▷), 하얀세모3(▽), . . . , 검은세모1(◀), . . ., ☏, ☎, ☜, ♨ 등과 같이 그룹핑할 수 있다. 역시 부속순위는 사용빈도를 고려하여 둘 수 있으며, 특정 버튼과 연관시키는 것은 적절한 방법을 택하면 된다. 단 앞에서의 0차원 형태의 기호, 1차원 형태의 기호를 연관시킨 버튼이 아닌 다른 버튼에 연관시키면 된다.

74> 위에서와 같이 0차원, 1차원, 2차원 형태의 3개 그룹(이를 편의상 "3대그룹"으로 부름)으로 기호를 그룹핑할 경우 연관되는 숫자버튼을 3개만 기억하면 되는 장점이 있으나, 빈번히 사용되지 않는 기호를 입력할 경우, 컨트롤버튼을 여러번 눌러야 하는 단점이 있다. 따라서 약간 그룹을 더 세분화하는 예(이를 편의상 "세부그룹"으로 부름)를 보이면 다음과 같다.

75> 먼저 2차원 형태의 기호그룹에서 line 의 결합형태를 이루는 기호그룹(예. *, #, %, ...)과, 단일폐곡선을 이루는 기호그룹(예. ◁, ◀, ...)으로 나눌 수 있다. 또한 그룹에서 ☏, ☎, ☜, ☞, ♨, ... 등과 같이 그림형태를 이루는 기호를 별도의 그룹으로 둘 수 있다. 역시 적절한 버튼에 속한 후속알파벳으로 두면 된다. 이들 그룹을 별도로 둘 경우, 앞의 기호그룹에서 이들 기호를 제외할 수도 있고 제외하지 않을 수도 있으며, 이는 다른 경우에 대해서도 마찬가지이다.

276> 다음으로 1차원 혹은 2차원 형태의 기호 중 수학식에 사용되는 기호들, 즉 +, -, *, /, root(√), sigma(∑), integral(∫), . . . 등을 별도의 그룹으로 둘 수 있다. 역시 적절한 버튼에 속한 후속알파벳으로 두면 된다.

277> 또한 방향을 나타낼 수 있는 기호들, 즉 →, ←, ↑, ↓, ↗, ↙, ↖, ↘, ◁, ▷, ◀, . . . 을 그룹으로 나눌 수 있다. 역시 적절한 버튼과 연관하여 후속알파벳을 간주하면 된다.

278> 그리고 비교적 타당성있게 또 하나의 그룹을 형성할 수 있는 각종의 괄호들, 즉 (,), [,], {, }, <, >, . . . 을 또 하나의 그룹으로 둘 수도 있다. 또한 괄호형태로 그룹핑함에 있어서도, 왼쪽 괄호와 오른쪽 괄호를 각각 그룹으로 둘 수도 있다.

279> 3대그룹만을 두지 않고, 세부그룹을 두고 그에 의하여 기호를 입력할 수 있도록 하는 경우는 세부그룹에 속한 알파벳을 3대그룹에 그대로 둘 수도 있고 두지 않을 수도 있다고 하였는데, 3대그룹에 두더라도 가능한 한 부속순위를 후순위에 두어야 할 것이다.

280> 이상의 기호 그룹핑을 도1-1에 적용하여 각 기호의 그룹을 특정버튼의 후속알파벳으로 간주하는 예를 보이면 다음과 같다. 0차원 형태의 기호그룹을 [0] 버튼에 속한 후속알파벳으로 간주, 1차원 형태의 기호그룹을 [1]버튼에 속한 후속알파벳으로 간주, 2차원 형태의 기호그룹을 [2]버튼에 속한 후속알파벳으로 간주, 2차원 형태의 기호그룹 중 단일폐곡선형태의 기호그룹을 [8] 버튼에 속한 후속알파벳으로 간주, 2차원 형태의 기호그룹 중 그림형태의 기호그룹을 [7]버튼에 속한 후속알파벳으로 간주, 수학식 기호그룹을 [6]버튼에 속한 후속알파벳으로 간주, 방향 기호그룹을 [3]버튼에 속한 후속알파벳으로 간주, 괄호 기호그룹을 나머지 숫자버

튼중 임의의 버튼에 속한 후속알파벳으로 간주할 수 있다. 설명한 바와 같이 기호 그룹을 각 버튼에 연관시킴에 있어서 이는 하나의 예일 뿐이며, 사용자의 편의대로 설정하도록 할 수 있다.

31> 이상을 종합하면, 기호를 특정버튼에 속한 후속알파벳으로 간주하여 컨트롤처리함에 있어서 3개의 그룹(0차원, 1차원, 2차원)으로 그룹핑할 수도 있으며, 혹은 약간 더 세분화하여 10개 이하의 그룹으로 그룹핑하여 거의 대부분의 기호를 입력할 수 있음을 보인 것이 본 발명의 핵심중의 하나이다. 또한 각 기호의 그룹을 특정버튼에 속한 후속알파벳으로 간주함에 있어서, 이름, 차원, 모양, 등등 기호의 그룹과 연관된 버튼에 속한 후속알파벳으로 간주하여 컨트롤처리함으로써, 기호를 키패드에 표시하지 않고, 간결한 알파벳 배치를 유지하면서, "은닉형 후속컨트롤처리방법"을 사용할 수 있는 것이 본 발명의 핵심중의 하나이다.

82> 본 출원에서 기호를 위한 컨트롤버튼으로 주로 [*] 버튼 혹은 상하좌버튼 중의 임의의 버튼을 사용하였는데, 숫자 및 영어알파벳을 위한 후속컨트롤버튼으로 [#] 버튼을 활용할 수 있다. 예를 들어 [#] 버튼이 이미 후속컨트롤 버튼으로 사용되고 있는 경우는 [#] 버튼상의 가용한 격자에 숫자 및 영어알파벳을 위한 컨트롤을 추가적으로 배치하면 된다. 이 경우도 컨트롤의 선택에 있어서, 앞에서 언급한 바와 같이, 대표알파벳과 결합할 수 없는 컨트롤은 건너뛰어 그 다음 가용한 컨트롤이 선택되도록 할 수 있다.

283> 8. 이동버튼 활용방법

34> 8.1 이동버튼의 컨트롤버튼으로의 활용

85> 선출원에서 각종의 컨트롤이 배치되는 컨트롤버튼을 4*3 키패드 내의 임의의 버튼에 둘 수도 있고, 4*3키패드 이외의 별도의 버튼에 둘 수도 있다고 하였다. 그리고 선출원에서 특히 알파벳의 갯수가 많고, 변형알파벳이 다수 존재하는 언어에 있어서, 4*3 키패드내에서 컨트롤 버튼으로 사용할 버튼이 부족한 것을 경험하였을 것이다. 본 발명에서는 문자입력모드에서 비교적 빈번히 사용되지 않는 좌/상/하 이동버튼의 1차기능을 선출원과 본 발명에서 언급하였던 컨트롤버튼으로 활용할 수 있음을 지적하고 그러한 예를 보인다. 즉, 문자입력모드에서 4*3 키패드 밖의 별도의 버튼을 컨트롤버튼으로 활용함에 있어서, 상/하/좌 이동버튼을 컨트롤 버튼으로 활용하게 되는 것이다.

86> 도8-1는 현재 전형적으로 사용되고 있는 폴더형 단말기의 버튼을 도시한 것이다. 파선으로 표시된 [i] 버튼은 인터넷 접속을 위한 버튼이며 이 버튼은 단말기에 따라서 존재하기도 하고 존재하지 않기도 한다. 여기서 우측 이동버튼은 공백입력용으로 사용되고, 특히 한국어에서는 1차모호성 제거를 위한 음절(글자)확정버튼으로도 사용되고 있다. 문자입력모드가 아닌 메뉴선택모드에서는 상/하/좌 이동버튼은 메뉴의 선택을 위한 이동버튼으로 유용하게 사용된다. 그러나 문자입력모드에서 상/하/좌 이동버튼은 사용빈도가 많지 않으며, 특히 상/하 이동버튼은 빈번히 사용되지 않는다. 편의상 상/하/좌 이동버튼을 합쳐 "상하좌 이동버튼" 혹은 "상하좌버튼" 이라고 부르기로 하고, 상/하 이동버튼을 "상하 이동버튼" 혹은 "상하버튼" 이라고 부르기로 한다. 어느 한 방향의 이동버튼만을 언급할 때에도 같은 요령으로 한다.

7> 8.2 이동버튼 하단 배치

8> 우선 이 상하좌버튼은 보통 숫자버튼 위에 배치되어 있는 경우가 보통이다. 그러나 문자입력모드에서 이들 버튼의 1차기능을 각종의 알파벳 입력을 위한 버튼으로 활용하려면, 주로 컨트롤버튼으로 활용되는 [*], [#] 버튼과 함께 4*3키패드의 하단에 배치되는 것이 유용함을 알 수 있다. 도8-2, 도8-3은 이를 도시한 것이다. 그러나 반드시 이렇게 해야 하는 것은 아니며, 편의상 본 발명의 예에서는 이동버튼을 4*3키패드 하단에 배치한 예로써 설명한다.

39> 여기서 4*3키패드와 상하좌버튼을 합하여 5*3키패드를 구성할 수 있음을 알 수 있다. 이는 부분전체선택방법의 적용시, 각 버튼을 3x5 의 15개 격자로 나누어 활용할 수 있는 것을 의미한다. 마찬가지로 반드시 도8-3에서와 같이 5*3키패드를 구성할 수 있도록 상하좌버튼을 배치하여야 하는 것은 아니다.

90> 8.3 이동버튼 좌측면 혹은 우측면 배치

91> 상하좌우 이동버튼을 4*3 키패드의 좌측 혹은 우측에 배치할 수도 있다. 이 경우 부분전체선택방법을 적용함에 있어서, 4*3 키패드와 이동버튼을 합하여 4*4 키패드를 구성할 수 있다. 도8-4는 상하좌우 이동버튼을 4*3키패드의 오른쪽에 구비한 경우의 예이다.

92> 이는 단말기에서 액정의 크기가 점점 커지는 추세에 있는데, 단말기의 액정의 크기를 더 크게 구성할 수 있는 장점이 있다. 또한 출원인의 한국특허출원 10-2000-0002081, 10-2000-0005671, 10-2000-0067852, 10-2001-0002137 에서 제시한 이동단말기에서의 측면배터리 장착방법을 병행적용시 좋은 특성이 있다.

3> 8.4 컨트롤버튼 및 알파벳입력을 위한 버튼으로의 각종 활용 사례

4> 다음에서, 이동버튼을 활용하는 사례를 든다. 단 아래의 사례에만 국한되는 것은 아니다.

5> 8.4.1 기호컨트롤버튼으로의 활용 사례

6> 한국어의 실시예에서, [*] 버튼과 [#] 버튼에 각각 격음컨트롤과 경음컨트롤을 배치한 경우가 있었다. 또한 [*] 버튼에 격음컨트로과 격음컨트롤을 배치하고, [#] 버튼을 기본모음컨트롤과 확장모음컨트롤만을 배치한 경우가 있었으며, [#] 버튼에 확장모음컨트롤만을 배치한 경우가 있었다. 이러한 경우, 기호컨트롤을 [*] 버튼 혹은 [#] 버튼상에 배치하면, 컨트롤버튼 후입력에 따라 먼저 격음, 경음 등이 먼저 나오고, 다음에 기호가 선택되게 된다. 이는 타 언어에 있어서도 마찬가지이다.

97> 그러나 기호컨트롤을 분리하여 상하좌버튼 중 임의의 버튼에 두게되면, 기호의 그룹과 연관된 버튼과 기호컨트롤이 배치된 기호컨트롤버튼을 조합하여 입력하는 즉시 기호를 입력할 수 있다. 도8-5은 하 이동버튼의 1차기능을 기호컨트롤의 선택을 위한 컨트롤버튼으로 활용한 예이다. 0차원형태의 기호그룹에 대하여 컨트롤 후입력을 적용한 예를 보이면, dot = [3]+[v], comma = [3]+[v]+[v], colon = [3]+[v]+[v]+[v], semi-colon = [3]+[v]+[v]+[v]+[v], . . . 과 같이 된다.

98> 8.4.2 한국어에서의 모음요소 버튼으로 활용 사례

9> 특히 선출원에서 한국어의 예에서 모음요소(ㅡ, ㅣ, .)을 활용한 경우에 있어서, 모음요소 "."과 "ㅎ"을 함께 배치함으로써, "ㅎ" 입력에 불편이 있었는데, 모음요소를 상하좌버튼 중 임의의 버튼에 배치함으로써, 이러한 불편을 극복할 수 있다. 만약 기호컨트롤을 하 이동버튼에 두고, 모음요소 "."을 상 이동버튼에 두면 상하좌버튼에서 2개의 버튼을 사용하게 된다. 혹은 상하좌버튼에 한국어의 모음요소 3개를 각각 둘 수도 있다.

)0> 격음컨트롤을 둔 버튼에 기호컨트롤을 둔 것과 마찬가지로, 상하좌버튼 중 임의의 버튼의 기준격자위치에 한국어의 모음요소 "." 를 두고, 기준격자에 가까운 순서로 기호컨트롤들을 둘 수도 있다. 도8-6은 이러한 예이다. 이 경우도 모음요소 "."은 단독으로 사용되지 않으므로, 반복선택방법으로 배치된 모음요소와 기호컨트롤을 선택하더라도 모호성없이 모음요소와 기호컨트롤을 선택할 수 있다. 도8-6에서 [v] 버튼을 1번 누르면 모음요소"."이 선택되고 두번 누르면, 기호컨트롤1이 선택되고, 3번누르면 기호컨트롤2가 선택된다.

01> 8.4.3 일본어에서 첨자컨트롤버튼으로의 활용 사례

02> 일본어의 경우에도, 50음도표의 배치를 각 버튼에 대응시키고, 2nd, 3rd 후속컨트롤을 [*] 버튼에 두고, 4th, 5th 후속컨트롤을 [#] 버튼에 둔 경우(선출원의 배정방법3), 장음/탁음/반탁음 입력을 위한 컨트롤을 상하좌버튼 중 임의의 버튼에 둘 수 있다. 도8-7을 참고할 수 있다.

03> 8.4.4 아랍어에서 모음 입력을 위한 첨자컨트롤버튼으로의 활용 사례

04> 아랍어의 경우 첨자형태의 모음을 처리하기 위한 컨트롤을 상하좌버튼 중 임의의 버튼(들)에 분산배치할 수 있다.

05> 8.4.5 태국어에 있어서의 컨트롤버튼으로의 활용 사례

06> 태국어의 경우도 자음컨트롤과 모음컨트롤을 분리하지 않고, 하나의 컨트롤버튼을 후속 컨트롤버튼으로 활용하였는데, 상하좌버튼 중 임의의 버튼을 컨트롤버튼으로 추가적으로 활용할 수 있다. 혹은 기타의 컨트롤버튼으로 활용할 수 있다.

07> 8.5 단축/풀 전환 컨트롤버튼으로의 활용 사례

08> 병행입력방법을 적용할 경우, 심플코드와 풀코드간에 발생하는 3차모호성이 있는데, 이를 없애기 위하여 단어단위의 "단축/풀" 전환 컨트롤을 두는 방법을 제시하였다. 예를 들어, 풀입력방법을 기본모드로 하고 병행입력방법적용시 단축입력방법에 의하여 단어를 입력하려면, 먼저 "단축/풀" 전환 컨트롤을 선택하고 공백(우 이동버튼)을 입력하고, 심플코드를 입력하는 것이었다. 물론 공백과 "단축/풀" 전환 컨트롤의 순서는 바뀌어도 된다. 여기서 단어단위의 "단축/풀" 전환컨트롤을 상하좌버튼 중 임의의 버튼에 둘 수 있다. 혹은 "단축/풀" 전환 컨트롤과 공백(우 이동버튼)을 합한 기능을 하는 컨트롤을 상하좌버튼 중 임의의 버튼의 기준격자 위치에 둘 수 있다. 도8-8을 참고할 수 있다.

309> 만약 도8-8의 숫자버튼에 도1-1의 영어 알파벳이 배치되어 있는 것으로 가정하고, 기본 알파벳입력모드를 풀입력모드인 상태에서 help의 전체연관 심플코드 "4357" 을 입력하여, 단어 help 를 입력하고자 할 경우, "~ 풀코드입력 +[^]+[4]+[3]+[5]+[7]+ [>]+ 풀코드 입력 ~" 과

같이 입력할 수 있는 것이다. 즉 상 이동버튼 [^]의 이동기능을 제한하고 단어단위의 "단축/풀" 전환 기능과 공백 기능을 합하였으므로 [^] 버튼이 눌려지는 순간 시스템은 그 다음 나오는 "[4]+..." 이 풀코드가 아닌 심플코드임을 인지하고 색인을 참조하여 입력 [4]에 가장 근접한 단어들을 사용자에게 추천하여 줄 수 있는 것이다. [4]+[3]+[5]+[7] 의 입력이 끝나고 공백 버튼([>] 버튼) 눌려지는 순간 단어가 종료되므로, 단어단위의 "단축/풀" 모드전환도 종료되고 시스템은 다시 풀코드의 입력을 기다리게 된다. 만약 심플코드(4357)의 입력후에 다시 [^] 버튼이 눌려지게 된다면, 시스템은 역시 단어가 종료된 것을 인지하여 심플코드 4357에 해당하는 help를 확정하고 다시 심플코드의 입력을 기다리게 될 것이다.

10> 병행입력방법에서 우 이동버튼(공백 버튼)을 하나만 사용하게 되면, 선출원에서 언급한 바와 같이 풀코드와 심플코드간에 발생하는 3차모호성이 있을 수 있고, 이를 회피하기 위하여 시스템은 버튼의 입력시 마다, 심플코드가 존재하는지 혹은 약속된 풀코드를 이루는지를 검사하여야 하였다. 그러나 이렇게 단어단위의 "단축/풀" 모드전환과 공백을 합한 버튼을 둠으로써, 입력값이 풀코드인지 심플코드인지 시스템이 미리 인지할 수 있으므로, 계산과 검색을 줄여, 시스템의 성능 향상이 가능할 것이다.

11> 8.6 계산기모드에서 가감승제 버튼으로의 활용 사례

12> 4개의 이동버튼은 그 위치에 상관없이 계산기모드에서 가장 빈번하게 쓰이는 가감승제 (+, -, x, /) 버튼으로 유용하게 사용될 수 있다. 가감승제 기호를 버튼에 표기할 수도 있으나, 알파벳입력에 비하여 계산기는 자주사용되지 않을 것이므로 버튼에 표기하지 않을 수도 있다. 알파벳입력모드에서는 각 버튼이 이동버튼 혹은 컨트롤버튼으로 사용될 것이기 때문이다.

3> 또한 계산기에서 쓰일 수 있는 연산자를 가감승제버튼에 두고 (은닉형) 반복선택방법에 의하여 선택하도록 하는 것이 가능하다. 계산기모드에서 자주사용되는 연산자(이항연산자)는 반복하여 출현하지 않는 성질을 이용한 것이다. 예를 들어 2++1 이라는 계산은 있을 수 없다. 또한 24 은 2xx4 로 풀어 승(x)버튼을 반복하여 누름으로써 "제곱"연산자를 선택하도록 할 수 있다. 즉 승(x)버튼에 후속연산자로 제곱(xx)연산자가 있는 것으로 하여 반복선택하는 것과 같다. 마찬가지로 (03)은 3//2 로 풀어 제(/)버튼을 반복하여 누름으로써 "제곱근"연산자를 선택하도록 할 수 있다. 이외의 이항연산자는 중복하여 출현하지 않으므로 적절한 가감승제버튼에 속한 후속연산자로 두고 후속연산자를 반복선택방법에 의하여 선택하도록 할 수 있다.

14> 상하좌우 이동버튼 중 3개의 버튼에 가감제(+, -, /) 연산자를 두고 승(x) 연산자는 [*] 버튼을 이용할 수도 있다.

15> 9. 도움기능 활성화

16> 여기서 각각의 입력모드에서 버튼(상/하/좌/우)에 표시되지 않은 기능을 스크린(액정)에 표시하여 사용의 편리를 더할 수 있다. 이 경우 액정의 일부를 소모하는 단점이 있는데, 물론 각 연산자버튼의 기능을 숫지하고 있는 사용자는 굳이 액정의 공간을 소비하면서 이를 표시되도록 할 필요가 없으나 그렇지 못한 사용자에게는 많은 도움이 될 수 있다. 도8-1을 참고할 수 있으며, 도9-1은 도8-4와 같이 상하좌우버튼이 4*3 키패드의 오른쪽에 배치된 경우를 가정하여, 유사한 형태로 배치한 경우의 예이다.

7> 이렇게 사용자의 편의와 의도에 따라 버튼의 기능(즉 버튼에 할당된 연산자 혹은 각 버튼에 연관된 기호그룹)을 표시하여 주는 것을 편의상 "도움기능 활성화"라고 한다. 도움기능 활성화는 각각의 모드(예. 알파벳입력모드, 계산기모드)에 대해서 이루어질 수도 있고, 각 모드에서 필요기능에 대해서(예. 알파벳입력모드에서 기호그룹에 연관된 숫자버튼 혹은 컨트롤버튼의 용도) 이루어질 수도 있다.

8> 마찬가지로 선출원에서 제시하였던 컨트롤버튼의 기능 혹은 기호그룹과 연관되는 숫자버튼 역시 사용자가 필요로 할 경우 액정에 표시하여 줌으로써 사용자의 편리를 도모할 수 있다. 도9-2을 참고한다. 도9-2는 앞에서 예시한 기호그룹분류에 의거하여 각 기호그룹에 연관되는 숫자버튼을 아이콘화하여 액정에 표시한 예이다. 각 숫자버튼에 연관된 기호그룹의 기호중 편의상 첫번째로 선태되는 기호만을 숫자버튼 아이콘에 부기한 것이다.

19> 10. 삭제버튼 활용방법

20> 삭제버튼을 활용함에 있어서 이를 선출원에서 소개한 "최종입력취소"로 활용할 수 있다. 예를 들어 도4-2에서 "ㄱ"을 입력하려다, [1]+[*]을 입력하여 "ㅋ"을 입력하였는데 삭제버튼을 눌러서 최종입력([*])을 취소함으로써 "ㄱ" 으로 환원시킬 수 있는 것이다. 이는 컨트롤버튼을 반복적으로 눌러서 후속알파벳을 입력하는 경우에 유용하게 사용될 수 있다. 취소버튼을 연속하여 누르는 경우는 통상의 방법대로 이미 입력된 알파벳이 삭제되도록하면 된다. 예를 들어 "가나ㅋ"을 입력한 상태에서 취소버튼을 1번 누르면 "가나ㄱ"이 되고, 한번 더 누르면, "가나"가 되고, 또 한번 더 누르면, "가" 가 된다. 로마자의 경우 "aba.."을 입력한 상태(마지

막의 "a.."은 "..+a"로 이루어진 변형알파벳)에서 삭제버튼을 한번 누르면 "aba"가 되고, 한번 더 누르면 "ab"가 되고, 또 한번 더 누르면 "a"가 되는 것이다. 즉 이미 형성된 알파벳에 대해서는 알파벳단위로 삭제되는 것이다.

21> 11. 키보드의 숫자키패드와 전화단말기에서의 키패드의 통일화

22> 선출원 및 본 발명에서 제시한 키패드는 이동단말기 혹은 표준키보드의 숫자키패드 혹은 스크린상에 소프트웨어적으로 구성되는 키패드 혹은 도어록 등 전화기자판 형태의 모든 분야에 응용될 수 있음은 자명하다. 또한 표준키보드에 구비된 숫자키패드는 선출원 및 본발명에서 제시한 키패드와 숫자 버튼의 배치가 다르나, 선출원 및 본 발명에서의 키패드 버튼상의 배치를 키보드에 구비된 키패드에도 적용할 수 있음은 자명하다. 예를 들어 선출원 및 본 발명에서 [1] 버튼에 배치된 알파벳을 키보드에 구비된 숫자 키패드의 [1] 버튼에 배치하고, 이하 마찬가지 방법으로 배치하여 알파벳입력, 심플코드의 활용 및 각종 코드의 암기용으로 활용할 수 있다.

323> 그러나 혼동을 줄이고 사용의 편리를 더하기 위하여, 키보드의 숫자키패드를 구성함에 있어서, 전화기 키패드의 숫자배치를 활용할 수 있다. 즉 키보드의 숫자키패드의 숫자배열에 있어서, 전화기의 키패드에서와 같이 제1행의 버튼에 [1], [2], [3] 버튼을 두고, 2행의 버튼에 [4], [5], [6] 버튼을 두고, 3행의 버튼에 [7], [8], [9] 버튼을 두는 것이다. 더불어 전화기의 키패드에서와 같이 [*] 버튼과 [#] 버튼을 둘 수도 있다.

<4> 12. 언어제한 입력방법

<5> 언어제한 입력방법은 특정언어의 단어생성에 있어서 자음과 모음의 결합규칙을 이용하여 문자입력에 있어서의 모호성을 줄이는 것을 의미하며, 이하에서 자세히 설명한다.

<6> 이하에서 어느 한가지 언어에서 언급한 내용중 다른 언어에서 적용할 수 있는 내용은 특별히 언급하지 않아도 다른 언어에 적용할 수 있음은 자명하다. 더 나아가 로마알파벳을 사용하지 않는 언어에 있어서도 유사한 개념을 적용할 수 있다.

<7> 12.1 로마알파벳을 사용하는 언어에 있어서 자모음분리키패드에서의 중국어제한 반복선택방법

<8> 12.1.1 중국어 성모와 운모의 구성

<9> 중국어에는 21개의 성모와 16개의 운모(이중 ㅖ (^e) 는 거의 사용되지 않는다고 함)가 있다고 한다. 성모는 자음에 해당하고, 운모는 모음에 해당한다고 한다. 중국대륙에서는 중국어음을 표기하는 방법으로 한어병음방안을 이용하고, 대만에서는 주음부호를 이용하여 중국어음을 입력한다고 한다. 한어병음은 라틴자모즉 로마알파벳을 이용하여 중국어음을 표기하는 것이다. 주음부호와 그에 대응되는 한어병음(로마알파벳)을 괄로안에 표시하면 다음과 같다.

0>

성모		ㄅ(b) ㄆ(p) ㄇ(m) ㄈ(f)
		ㄉ(d) ㄊ(t) ㄋ(n) ㄌ(l)
		ㄍ(g) ㄎ(k) ㄏ(h)
		ㄐ(j) ㄑ(q) ㄒ(x)
		ㄓ(zh) ㄔ(ch) ㄕ(sh) ㄖ(r)
		ㄗ(z) ㄘ(c) ㄙ(s)
운모	단운모	ㄚ(a) ㄛ(o) ㄜ(e) ㄝ(ê) (i) ㄨ(u) ㄩ(ü)
	복운모	ㄞ(ai) ㄟ(ei) ㄠ(ao) ㄡ(ou)
	부성운모	ㄢ(an) ㄣ(en) ㄤ(ang) ㄥ(eng)
	권설운모	ㄦ(er)

31> 다음은 중국어의 결합운모를 설명한 것이다. 결합운모는 3가지 모음 i, u, ..u 를 앞세우고 뒤에 다른 운모가 결합하는 것이라고 한다. 다음 표는 결합운모 일람표로 결합이 가능한 경우를 표시한 것이라고 한다.

32>

	단운모			복운모				부성운모			
	a	o	e	ai	ei	ao	ou	an	en	ang	eng
i	O		O			O	O	O	O	O	O
u	O	O		O	O			O	O	O	O
¨u			O					O	O		O

33> 위 표에서 ia 의 결합은 가능하나, io 의 결합은 불가능한 것을 알 수 있다.

34> 12.1.2 로마알파벳을 이용한 중국어 한어병음의 입력

35> 결국 위의 표에서 알 수 있듯이, 로마알파벳을 이용하는 경우, 중국어의 21개 성모는 18개 로마알파벳의 조합으로 모두 입력할 수 있고, 16개 운모는 7개의 단운모의 조합 혹은 로마알파벳 모음과 로마알파벳 자음의 조합으로 모두 입력할 수 있게 된다.

6> 또한 단운모 중 ^e 와 ..u 는 각각 e 와 u 의 변형알파벳으로 간주하여 컨트롤처리방법에 의하여 입력할 수 있다. 다음으로 5개의 기본모음에 4가지 성조부호가 붙을 수 있는데, 이 역시 컨트롤처리방법에 의하여 입력될 수 있다. 결국 로마알파벳을 사용하는 중국어의 경우 5개의 로마알파벳 기본모음을 이용하여 모든 중국어의 운모를 입력할 수 있는 것이다. 이는 영어알파벳(즉, 로마알파벳)에 존재하지 않는 변형알파벳을 컨트롤처리방법에 의하여 입력하는 것으로 프랑스어, 독일어 등의 입력에서 이미 보인 내용이다. 중국어의 운모를 컨트롤처리방법에 의하여 입력할 경우의 기본알파벳(기본모음)과 후속알파벳(후속모음)과의 부속관계를 설정하는 예를 보이면 다음과 같다.

37>

	기본모음	후속모음				
		2nd	3rd	4th	5th	6th
1 그룹	ㄚ (a)	ā	ˊa	ǎ	ˋa	
2 그룹	ㄛ (o)	ˉo	ˊo	ǒ	ˋo	
3 그룹	ㄜ (e)	ē	ˊe	ě	ˋe	ㄝ (ê)
4 그룹	(i)	ī	í	ǐ	ì	
5 그룹	ㄨ (u)	ū	ˊu	ǔ	ˋu	ㄩ (ü)

38> 위의 예는 1성조 ~ 4성조의 성조부호가 붙은 알파벳이 후속알파벳으로 되었고, ^e 는 거의 사용되지 않는다고 하므로 부속순위를 가장 하위에 둔 것이다. 부속순위는 사용빈도 등에 따라 달라질 수 있음은 물론이다. 예를 들어 ..u 를 6th 가 아닌 2nd 후속알파벳으로 하고 나머지 후속알파벳의 부속순위를 하나씩 후 순위로 둘 수도 있다.

39> 입력예로 컨트롤버튼으로 [*] 버튼을 사용하고, 컨트롤 후입력 적용시, ¯e = e + [*], ˊe = e + [*]+[*], ˇe = e + [*]+[*]+[*], ˋe = e + [*]+[*]+[*]+[*], ^e = e +

[*]+[*]+[*]+[*]+[*] 과 같이 된다. 도1-1에서 키패드상 표시된 "e" 를 부분전체선택방법을 적용하면, e = [3]+[2] 가 되므로 -e = [3]+[2] + [*] 가 된다. 도1-1에서 키패드상 표시된 알파벳을 부분전체선택방법으로 선택하면, 人(r/en : 제2성) 의 풀코드는 "7832**65" 가 된다. 여기서 키패드상 표시된 "e" 를 입력하는데 있어서 부분전체선택방법이 아닌 어떠한 다른 입력방법(예. 단순반복선택방법)도 사용될 수 있음은 자명하다.

40> 결국 컨트롤버튼을 단 하나만 사용하므로 입력방법을 단순화 할 수 있다. 이는 로마알파벳을 사용하지 않고 주음부호를 사용하는 경우에도 동일하게 적용할 수 있다(예를 들어 도1-1에서 영어알파벳에 대응되는 주음부호를 표시한 키패드에서 동일하게 입력방식 적용).

41> 12.1.3 자모음분리키패드에서의 중국어제한 반복선택방법

42> 중국어 성모를 18개 로마알파벳으로 모두 입력할 수 있는데, 그중 zh, ch, sh 만 로마알파벳 자음이 연속하여 나온다. 중국어에서 성모없이 운모만으로 이루어진 음절을 한어병음으로 표기할 때, 형식적으로 y, w 등을 붙여 표시한다고 한다. 예를 들어 "衣 = yi(1성)", "五 = wu(3성)" 와 같이 된다.

43> 그리고 선출원에서 언급하였듯이, 한자(漢字) 한글자가 한 음절을 이루는 중국어에 있어서 음절기준 이니셜코드를 심플코드로 사용하는 것이 많은 편리를 줄 수 있다. 그러므로 가능한 18개의 로마알파벳 자음이 각 숫자 버튼에 배정되는 것이 바람직하다. 본 발명에서는 18개 로마알파벳 자음을 2개씩 그룹핑하여 [1] ~ [9] 숫자 버튼에 배정하는 것을 제시한다.

44> b p / d t / g k / z j / c q / s x / m n / l r / h f

5> 위의 예는 발음의 유사성에 근거하여 그룹핑한 사례로, 심플코드의 활용시 동일한 심플코드(음절기준 이니셜코드)에 대응하는 어구(단어 혹은 구절)가 다수인 경우 발음이 유사한 어구가 같은 음절기준 이니셜코드를 가지도록 배려한 것이다. 위의 예는 단지 예시중 한가지일 뿐이며, 많은 변형이 가능하다. 발음의 유사성에 의하여 그룹핑할 수도 있으며, 영어알파벳 사전순서, 영어알파벳 대응 주음부호 사전순서 등 다양한 기준에 의하여 그룹핑이 가능하다. 이렇게 유사한 발음을 같은 그룹으로 그룹핑하여 같은 버튼에 배정하는 또 다른 잇점은 로마알파벳을 사용하는 모든 언어에 있어서, 유사한 발음의 자음이 연속하여 나오는 경우가 적으므로, 그 만큼 모호성이 발생할 가능성이 적어지는 것이다. 또한 심플코드(특히 음절기준 이니셜코드)의 활용에 있어서도 동일한 심플코드에 대응되는 어구가 다수 있더라도, 그 어구들이 유사한 발음(음가)를 가진 어구들이므로 사용자에게 혼란을 최소화하고, 심플코드 활용에 자연스러움을 줄 수 있다.

46> 또 한가지는 zh, ch, sh 가 중국어의 성모로써 존재하는데, z 와 h 가, c 와 h 가, 그리고 s 와 h 가 같은 그룹으로 그룹핑되지 않도록 한 것이다. (중국어에서 보통 "성모+운모"로 음절이 이루어 지는데 본 실시예에서 같은 그룹으로 그룹핑되어도 큰 상관은 없다. 단, 도10-1에서 표시되지 않은 w, y, v 등을 입력시 s, h, w 등을 함께 배정하고 배정된 버튼 3타로 입력하도록 하지 않으면 된다)

47> 위에서 예시한 9개 그룹을 도10-1에서와 같이 [1] ~ [9] 의 9개 버튼에 임의로 배정하고, 각 알파벳의 입력에 반복선택방법을 적용할 수 있다. 중국어의 특성상 성모를 입력시 zh, ch, sh 의 경우를 제외하고는 로마알파벳 자음이 연속하여 나오는 경우는 없게된다. 따라서 반복선택방법에 의하여 알파벳을 입력하더라도 모호성 없이 입력이 가능하다. 예를 들어 도

10-1에서 b = [1], p = [1]+[1] 로 입력이 가능한 것이다. 각 버튼에 배정된 알파벳 중 어느 알파벳을 1타로 선택되도록 할 것인지는 알파벳의 사용빈도에 따라 사용빈도가 많은 알파벳이 1타로 선택될 수 있도록 정해질 수 있음은 물론이다.

48> 중국어에서 거의 사용되지 않는 로마알파벳 "v" 를 9개의 그룹중 임의의 그룹에 둘 수 있다. 예를 들어 발음이 유사한 "f" 가 있는 그룹에 "v"를 추가로 두고(혹은 명시적으로 배정하지 않고) 해당 버튼 3타로 "v"를 입력하도록 할 수 있다. "衣 = yi(1성)", "五 = wu(3성)" 등에서 사용되는 w, 와 y 역시 적절한 자음 그룹에 두고(혹은 명시적으로 배정하지 않고) 이를 반복선택방법에 의하여 입력(예. w와 y를 각 알파벳이 속한 버튼 3타로 입력)하도록 할 수 있다. 예를 들어 y 를 / l, r / 이 속한 그룹에 두었을 경우, 해당 버튼(도10-1에서 [8] 버튼) 1타로 “l” 을 입력하고, 2타로 “r” 을 입력하고, 3타로 “y” 를 입력하며, w 를 / m, n / 이 혹한 그룹에 두었을 경우, 해당 버튼(도10-1에서 [7] 버튼) 1타로 “m” 을 입력하고, 2타로 “n” 을 입력하고, 3타로 “w” 를 입력하는 것이다.

49> 반복선택방법을 적용하여, 한어병음 입력시 로마알파벳 자음과 모음이 번갈아 나오는 성질을 이용하여 모호성을 대폭 줄이는 것을 편의상 “중국어제한 반복선택방법” 이라 부른다. 이러한 성질을 특별히 중국어뿐만이 아닌 모든 언어에 일반적으로 적용시에는 편의상 “언어제한 반복선택방법(LRRSM : Language Restricted Repeat Selection Method)” 이라 하고, 특별히 중국어에 적용하였을 경우를 “중국어제한 반복선택방법(Chiese Restricted RSM)” 이라 한다. 언어제한 반복선택방법은 한국어 혹은 힌디어 등에 있어서, 기본자음과 기본모음의 쌍을 각 버튼에 배정하고 반복선택방법을 적용시, 자음과 모음이 번갈아 나오는 성질을 이용하여 적은 모호성으로 입력할 수 있도록 한 것과 같은 맥락이다. 또한 한국어의 모음요소를 활용한 방법에

있어서, 모음 "—" 가 연속하여 나오지 않는 성질을 이용하여 격음컨트롤을 반복선택방법에 의하여 선택할 수 있도록 한것도 유사한 것이며, 이는 "한국어제한 반복선택방법" 이라 할 수 있다.

<50> 특히 도10-1 ~ 10-4에서와 같이 자음이 배정된 버튼(편의상 "자음버튼" 이라 부름)과 모음이 배정된 버튼(편의상 "모음버튼" 이라 부름)을 분리하고(즉, 자음과 모음을 같은 버튼에 배정하지 않고), 반복선택방법을 적용시 자음과 모음이 결합하는 각 언어의 특성을 이용하여 모호성을 대폭 줄일 수 있는 좋은 특징이 있게 된다. 이렇게 도10-1 ~ 10-4에서와 유사하게 자음버튼과 모음버튼을 각각 분리하여 각 버튼에 소정 개수(예. 1~3개)의 자음 혹은 모음을 배정하여 구성한 키패드를 편의상 "자모음분리키패드(CVSK(Consonant Vowal Separated Keypad))" 라 부르기로 한다.

<51> 도10-1에서 키패드상 표시된 로마알파벳 자음과 모음을 반복선택방법에 의하여 중국어 한어병음을 입력시에도 모호성없이 입력할 수 있는 것은 쉽게 알 수 있다. 성모의 입력을 위하여 자음버튼(도10-1에서 [1] ~ [9] 버튼)중 같은 버튼이 연속적으로 눌러졌을 때, 키패드상 표시된 2번째 자음이 입력된 것을 시스템이 쉽게 알 수 있다. 중국어 성모를 입력하기 위하여 같은 로마알파벳이 연속하여 나오는 경우가 없기 때문이다. 여기서 기 2번 눌려진 버튼이 w, y, v 등이 배정된 버튼이고, 같은 버튼이 한번 더 눌러졌을 때(즉 3번 눌러졌을 때)는 기 설명한 바와 마찬가지로, w, y, v 등을 입력하기 위한 것을 시스템이 쉽게 알 수 있다. [7] 버튼 3타로 "w = 777" 로 입력할 경우, 이는 "mmm" 혹은 "mn" 혹은 "nm" 으로 해석될 수 있는데, 중국어의 성모가 이와 같이 구성되는 경우는 없기 때문이다.

52> 도10-1에서와 같이, a, e, i, o, u, ..u 의 6개의 모음을 2개씩 3개의 그룹으로 나누어 4*3 키패드의 3개 버튼(예. [*], [0], [#] 버튼)에 두고 각각의 모음을 반복선택방법에 의하여 모호성없이 입력할 수 있다. 이것이 가능한 이유는 중국어에 있어서 동일한 로마알파벳 모음이 2번 연속 나오는 경우는 없기 때문이다. 예를 들어 어떤 한자를 한어병음으로 표기할 때, "…aa..." 과 같이 되는 경우는 없기 때문이다.

53> 단 여기서 복운모 ai, ei, ou 와 결합운모의 ia, ie, uo 를 놓고 보면, a 와 i, 그리고 e 와 i 그리고 o 와 u 는 같은 그룹으로 그룹핑되지 않아야 한다는 것을 알 수 있다. 만약 a 와 i 가 같은 그룹으로 그룹핑되어 같은 버튼에 배정되면(예. [*] 버튼), 해당버튼 3타(즉, '***')를 입력한 것이 'ai'를 입력한 것인지 'ia'를 입력한 것인지 알 수 없는 모호성이 발생할 수 있기 때문이다. 도10-1은 이러한 내용을 고려하여 그룹핑한 사례를 보여준다. 도10-1의 모음 그룹핑 및 배치가 절대적인 것은 아니고 위 제약 사항을 만족하는 한에서 변형이 가능함은 물론이다. 도10-1에서 각 모음을 반복선택방법에 의하여 입력할 수 있다. 예를 들어 i = [0], o = [0]+[0], iao = [0]+[*]+[0]+[0] 와 같이 입력할 수 있다.

54> a 와 i, 그리고 e 와 i 그리고 o 와 u 는 같은 그룹으로 그룹핑되지 않는 한에서, "a, e, o" 를 하나의 그룹으로 하고(도10-2, 도10-4 참고), 나머지 모음 "i, u, u" 를 하나의 그룹으로 할 수 있다. 역시 다양한 변형이 가능할 것이다. 이 2개의 모음 그룹을 각각 3*4 키패드 내에서 임의의 버튼(예. 각각 [*], [#] 버튼)에 할당하고, 자음 "w, y, v" 를 별도의 그룹으로 나머지 버튼(예. [0] 버튼)에 할당할 수 있다. 하나의 그룹에 3개의 알파벳이 그룹핑될 때 (예. "a, e, o"), 모음의 조합(예. "ae", "ea", "aaa")을 나타내는 해당버튼 3타에 해당하는 중국어 운모가 존재하지 않으므로, 해당버튼 3타로 "o"가 모호성 없이 인식될 수 있다. 예에서 "ao" 입력시 해당버튼 4타로 되는데, "oa" 가 한어병음으로 존재하지 않으므로 역시

모호성 없이 인식될 수 있다. 도10-6을 참고할 수 있다. 도10-6에서 "v" 는 중국어의 입력에 사용되지 않으므로 표시하지 않을 수도 있으며, "w, y" 등을 하나의 그룹으로 그룹핑하여 같은 버튼에 배정하는 것은 반자음(즉, 반모음)을 같은 버튼에 배정하는 효과가 있다.

<55> 도10-2, 10-4, 10-6에서 "a, e, o" 가 같은 그룹으로 그룹핑 되었는데, "ao" 의 조합만이 가능하다. 만약 하나의 버튼에 "a, o" 만이 배정된다면, 해당 버튼 3타는 "ao" 로 모호성없이 인식될 수 있을 것이다. 그러나 3개의 알파벳이 배정되고 버튼 누름의 횟수에 따라 "a ? o ? e" 의 순서로 선택된다면, 해당 버튼을 3번 누른 것이 "ao" 인지 "e" 인지 알 수 없는 모호성이 발생할 수 있다. 따라서 이러한 경우(3개 이상의 모음이 같은 버튼에 배정되고 3개의 모음 중 2개 모음의 한가지 조합이 가능한 경우)에 조합이 가능한 2개 모음을 해당 버튼 각각 1타, 2타로 선택되도록 하지 않고, 각각 1타, 3타(혹은 2타, 3타도 가능하며 해당 버튼 5타를 "ao" 로 인식할 수 있음)로 선택되도록 함으로써, 언어제한을 이용하여 모호성을 제거할 수 있다. 정리하면, 3개 알파벳 그룹에서 버튼누름 횟수에 따른 선택순서를 정함에 있어서, 연속하여 등장할 수 있는 알파벳을 각각 해당버튼 1타, 2타로 선택되지 않도록 선택순서를 정하는 것이다.

<56> 도10-1 ~ 도 10-4 에서 로마알파벳으로 성모를 입력시, 같은 버튼이 최대 2~3 번 눌러질 수 있으며, 시스템은 버튼의 반복누름 횟수를 인식하여, 그 반복누

름횟수에 따라 타겟알파벳을 식별해낼 수 있다. 로마알파벳으로 운모입력시, 역시 도 10-1, 도10-3 에서 같은 버튼이 연속하여 3번 눌러질 수 있는 경우는 결합운모 "iou" 의 "io" 를 입력하기 위하여 [0] 버튼이 연속하여 3번 눌러지는 경우뿐이다. 사용자의 입력에 대하여 시스템은 "0 = i" 로 일시적으로 인식하고, [0] 버튼이외의 버튼이 눌러지면 확정한다. [0] 버튼이 한번 더 눌러지면, 시스템은 "00 = o" 로 일시적으로 인식하고, 마찬가지로 [0] 버튼 이외의 버튼이 한번 더 눌러지면 확정한다. 여기서 [0] 버튼이 한번 더 눌러지면 시스템은 "000 = io" 로 확정할 수 있다. 왜냐하면, 한어병음의 입력에 있어서 "oi" 가 나오는 경우는 없는 "중국어제한" 을 이용하기 때문이다. [*] 버튼의 경우는 최대 2번 눌러질 수 있으므로, 시스템은 "* = a" 로 일시적으로 인식하고, "** = e" 로 확정할 수 있다. 한어병음 입력시 동일한 로마알파벳 모음이 연속하여 나오지 않고, 같은 버튼에 배정된 모음이 연속하여 나오는 경우가 없기 때문이다.

<57> 자모음분리키패드에서 모호성없이(혹은 중국어 이외의 경우에서 현저히 적은 모호성으로) 입력시스템의 구성이 가능한 이유는, 같은 자음버튼(예. 도10-1에서 [1] 버튼)이 한번 혹은 연속하여 눌러지는 동안 시스템은 동일 버튼내의 자음 중의 하나를 입력하는 것으로 인식하게되고, 뒤이어 다른 자음버튼(예. 도10-1에서 [9] 버튼)이 눌러지면 시스템은 연속된 자음이 입력되는 것으로 인식하게 되기 때문이다. 또 한가지 이유는 각 버튼당 약 3개의 알파벳을 배정하지 않고 약 2개의 알파벳이 배정되기 때문이다. 자음과 모음을 쌍으로 그룹핑하는 경우의 선출원에서 언급한 바, 버튼당 3개의 알파벳이 배정되었을 때에 비하여 2개의 알파벳이 배정되었을 때, 모호성 발생 가능성은 현저하게 줄어들며, 이는 쉽게 알 수 있다. 또한 버튼당 2개의 알파벳이 배정되었을 때는 출원인이 제시한 연타지연시간과 이타지연시간을 더욱 효과적으로 적용할 수 있다.

<8> 도10-7는 (R) 부분을 제외하면, 자모음분리키패드에서의 일반적인 반복선택방법의 절차를 나타낸 것이다. 도10-7에서 편의상 입력값은 자음버튼 혹은 모음버튼인 것으로 한다. 즉, 특별한 용도 혹은 기능(예. 후속알파벳의 입력을 위한 후속컨트롤버튼)은 없는 것으로 가정한다. 도10-7은 절대적인 것은 아니며 단지 참고할 수 있다. 도10-7에서 특정버튼이 한번 눌러짐에 대하여 특정버튼의 1회 누름에 대응되는 알파벳을 타겟알파벳으로 일시적으로 인식하는 등의 세부적인 절차는 표시하지 않고 있다. 다만 이러한 과정은 도10-7에서 (R0)과정에서 수행될 수 있다. (R)과정에서 언어제한을 판단할 수 있으며, (R)과정은 흐름도상의 임의의 부분에서 개입될 수 있다. 즉 첫번째 입력값이 자음버튼들 중의 하나이고, 연속 눌려진 버튼에 자음이 2개만 배정된 버튼이고, 같은 버튼에 배정된 알파벳이 연속하여 나오는 경우가 없으면(예. 도10-1에서 [1] 버튼), (R1) 에서 동일버튼이 2번 눌려진 순간 입력값을 하나의 자음(예. 도10-1에서 "P" 로)으로 확정할 수 있는 것이다. 이는 마치 도4-4에서 언어제한을 이용하여 모호성을 제거하는 것(반복선택방법 적용시 모호성없이 혹은 적은 모호성으로 타겟알파벳을 확정)과 유사하다. 도10-7의 절차는 도4-4의 절차보다는 훨씬 단순한 것을 알 수 있다.

<59> 동일한 혹은 동일하지 않은 자음버튼들이 연속하여 눌러지다가, 모음버튼이 눌러지면 시스템은 기 입력된 자음버튼들의 입력값으로부터 타겟알파벳을 확정하고, 모음처리과정을 진행하며, 모음처리과정에서의 최초 입력값은 자음처리과정에서 마지막으로 입력된 모음버튼이 된다. 모음처리과정에서 자음처리과정으로 바뀌는 경우에도 마찬가지이다.

<60> 도10-7에서 언어제한 (R) 에 의하여 판단하는 방법은 결합가능한 자음 및 모음의 목록(예. "ch", "sh", "zh", "iao", "iou", "ia", "ie", "uai", "uei", "ua", "uo", "ue", "ai", "ao", "ua", "uo", "..ue", "ou" …) 혹은 (특정 키패드에서 모호성이 발생할 수 있는 경우의) 결합불가능한 자음 및 모음의 목록(예. "bb", "aa",

"ee", "oa", …)을 시스템이 가지고 있으면서, 입력값이 모호성을 일으킬 수 있는 경우에, 결합불가능한 경우를 제외하고 결합가능한 경우를 시스템에서 타겟알파벳으로 인식하도록 하는 방법이 방법이 일반적으로 가능하다. 예를 들어, 도10-2(중국어제한)에서 "ao = 0000" 을 입력시, 시스템은 "ao"를 모음결합가능 목록에 가지고 있으면서, 시스템은 입력값 "0000" 에 대하여 "ao"를 타겟알파벳으로 인식할 수 있다. 또 한가지 방법은 도10-2, 도10-4의 자모음분리키패드에서 사용자의 오입력이 없을시, [0] 버튼 만이 최대 4번 눌러질 수 있는데, 이렇게 모호성이 발생할 수 있는 특별한 경우만 고려하여, 이 경우 "0000 = ao" 로 인식되도록 할 수 있다. 나머지 과정은 도10-7의 처리과정에 따르면 된다.

<31> 도10-7에서 생략된 부분의 절차는 기 표시된 절차와 유사함을 알 수 있다. 도10-7을 더욱 일반화하여 생략된 부분 없이 표현하면 도10-8과 같다. 도10-8은 도10-7과 같은 의미이다. 도10-8에서 {1} 부분은 버튼입력이 이루어짐을 가상적으로 의미하며, (1) 부분으로 들어오는 화살표에 표시된 "n <- (n+1)" 은 버튼누름의 횟수가 하나 더 증가함을 의미한다.

<62> 중국어 음절을 병음으로 표기함에 있어서, "en", "eng", "an", "ang", "er" 등의 운모가 사용되는 경우, 음절의 끝이 영어자음(영어알파벳 자음)으로 끝나게 된다 (예. ren : 人). 즉 단어 혹은 구절의 중간에서 어느 음절의 끝에 올 수 있는 영어 알파벳 자음은 "n", "g", "r" 등인 것이다. 이렇게 병음의 어느 음절의 끝에 올 수 있는 자음들을 편의상 "병음음절끝가능자음(CCESP : Consonants which Can be at the End of a Syllable of Pinyin)"이라 부르기로 한다.

<63> 이러한 병음음절끝가능자음이 사용되는 음절(즉, "en", "eng", "an", "ang", "er" 등의 운모가 사용되는 음절)의 경우, 대표적으로, 다음 음절의 영어자음이 앞 음절의 영어자음인 경우 모호성이 발생할 수 있게 된다. 예를 들어, Zhongguo (中國) 의 경우,

"~gg~" 가 "~k~" 로 인식될 수 있는 것이다. 이렇게 매우 특별한 경우에 모호성이 발생할 수 있는데, 이는 병음음절끝가능자음(예. "n", "g", "r" ...)을 이들이 속한 각각의 그룹에서 해당 버튼 2타로 선택되도록 하면 모호성을 제거할 수 있다 (도면10-6에서와 같이 "n", "g", "r" 등이 속한 그룹에 2개의 알파벳만이 배정되어 있을 때).

64> 도면10-6에서 병음음절끝가능자음 중의 하나인 "n"이 해당버튼 ([7] 버튼) 2타로 선택되므로 "rennai (忍耐)"의 경우 "88**7777*#"로 모호성없이 입력될 수 있는 것과 같다. "nn"을 입력하기 위하여 "7777"을 입력하게 되는데, 이는 모호성없이 "nn"으로 인식될 수 있는 것이다 (병음에서 단어 혹은 구절의 중간에 "mmmm", "nmm", "mnm", "mmn" 등의 조합이 불가능하므로). 즉, 도면10-6에서 "g"와 "k"의 선택순서를 "k - g" 로 하면 병음을 모호성없이 입력할 수 있는 것이다.

65> 이렇게 병음음절끝자음을 해당 그룹에서 2타로 선택되도록 함으로써, 모호성을 제거할 수 있으려면, 해당 그룹에 2개의 알파벳이 그룹핑되어 있어야 한다. 만약 도면10-4에서와 같이 3개의 영어알파벳 "m, n, w" 가 그룹핑되어 있고 버튼의 누름횟수에 따라 "m - n - w" 의 순서로 선택된다고 하면, "n"이 해당버튼 2타로 선택된다고 할지라도, "renmin (人民)"를 입력시 "rewin" 와 모호성을 일으키게 될 것이다 (여기서 "rewin"이 실제로 중국어 병음에서 존재하지 않으므로, 시스템이 입력값을 "renmin (人民)"으로 인식할 수도 있지만, "rewin"이 가능한 병음 조합이므로).

366> 따라서 도면10-4에서와 같이 영어자음이 3개씩 그룹핑되는 것이 가능한 경우, 병음음절끝가능자음 ("n", "g", "r" 등)은 2개의 영어자음들이 그룹핑되는 그룹으로 그룹핑할 필요가 있음을 알 수 있다. 그리고 자음이 2개씩 그룹핑되는 경우에도, 병음음절끝자음 중 "n" 과 "r", 그리고 "g" 와 "r" 이 같은 그룹으로 그룹핑되는 경우 역시 모호성이 발생할 수 있는 것을 알

수 있다. 예를 들어 "n"과 "r" 2개의 영어자음이 같은 그룹으로 그룹핑되어 있으면, "~nr~" 입력시 "~rn~"과 모호성을 일으키게 될 것이다. "g" 와 "r" 이 같은 그룹으로 그룹핑되는 경우도 마찬가지이다.

7> 도면10-*에서 "n"과 "g"는 각기 다른 버튼에 배정되어 있으므로 모호성없이 입력될 수 있다. 물론 "n"과 "g"가 같은 그룹에 그룹핑되어 있더라도, "n"과 "g"만이 그룹핑된다면, 모호성없이 입력될 수 있다. 예를 들어 "n" 과 "g"가 같은 그룹으로 그룹핑되어 [5] 버튼에 배정되어 있다면, "모음+ng" 를 입력하기 위하여 [5] 버튼이 3번 눌러지면(즉 "555"가 입력되면), 이를 "모음+gn" 이 아닌"모음+ng"로 인식할 수 있다. 병음에서 단어의 중간에 "모음+gn"이 나올 수 없는 언어제한을 이용할 수 있기 때문이다. 이렇게 "n", "g"가 같은 그룹으로 그룹핑되어 있을 경우, 버튼누름횟수에 따른 선택순서를 "g - n" 로 하면, 해당버튼을 2번눌렀을때, "g"가 표시되고, 한번 더 누르면, "ng"가 표시되도록 할 수 있으므로 자연스럽게 WISWYG (What You See is What You Get, 더 정확히는 What You Press is What You See) 를 실현할 수 있게 된다. 도10-2, 10-4, 10-6에서 모음 "a", "e", "o"가 하나의 그룹으로 그룹핑되어, [*] 버튼(도면10-6의 예)에 배정되어 있는데, 해당 버튼의 누름횟수에 따른 선택순서를 "o - e - a" 로 하면, 역시 [*] 버튼이 3번눌러졌을 때, 시스템은 "a"로 인식하여 표시하여 줄 수 있고, 다시 [*] 버튼이 한번 더 눌러지면 자연스럽게 "ao"를 표시하여 줌으로써 사용자에게 더욱 친화감을 줄 수 있다.

68> "n"과 "g"만이 같은 그룹으로 그룹핑되어 어느 버튼(예에서 [5]버튼)에 배정되어 있을 때, 모음버튼이 눌러지고 나서 "5555"가 입력된다면, "모음+ngg" 로 인식되고, "55555"가 입력된다면, "모음+ngn"으로 인식될 것이다. 중국어 병음에서 "모음+ng"가 비교적 많이 사용된다면, 이렇게 "n" 과 "g" 를 같은 그룹으로 그룹핑함으로써, 입력의 편리도를 높일 수 있다.

그러나, 심플코드의 활용을 위하여 유사한 발음을 가지는 자음을 같은 그룹으로 그룹핑하는 것이 자연스러우므로 좋은 접근방법을 아니라고 할 수 있다.

<69> "n" 과 "g" 이외의 영어자음(예. 'm')이 함께 그룹핑될 경우, 병음을 모호성없이 입력하기 위하여, "m - n - g" 혹은 "m - g - n" 의 순서로 선택되도록 하면(즉, 'n', 'g' 이외의 자음을 해당버튼 1타로 선택되도록 하면) 역시 모호성없이 인식될 수 있다. 예를 들어 버튼누름횟수에 따라 "m - g - n" 의 순서로 선택시, 모음버튼이 눌러지고 나서, 해당버튼을 5번 누른 것은 "(모음)+ng"로 인식되고, 6번 누른 것은 "(모음)+ngm"으로, 7번 누른 것은 "(모음)+ngg"로 인식되고, 8번 누른 것은 "(모음)+ngn"으로 모호성없이 인식될 수 있다. 그러나 "ng" 입력시의 반복누름 횟수가 과다하게 많아지므로 바람직하지 않을 것이다.

<70> 도면10-7, 10-8. 10-9 에서, 단어의 처음이 아닌 단어의 중간에서의 자음처리과정에서 "nX", "ngX", "rX" (대문자 "X" 는 성모를 영어자음으로 표시한 것을 의미함) 와 같이 자음이 연속하여 나오는 경우도, 병음에서 가능한 자음조합으로 입력값을 처리하면 된다. 단어의 처음에는 "nX", "ngX", "rX" 와 같이 영어자음이 연속하여 나오는 경우는 없기 때문이다. 결국 병음의 입력에서, 단어 혹은 구절의 중간에 나올 수 있는 영어자음은 이론적으로 "~ngch~", "~ngsh~", "~ngzh~" 와 같이 최대 4개이며, 도면10-7, 10-8, 10-9 에서 단어의 중간에서의 자음처리과정에서 이러한 언어제한을 이용하면 된다.

<71> 음절의 끝에 영어자음이 옮으로써, 모호성이 발생할 수 있는 경우에도 모호성없이 입력할 수 있는 방법을 제시하였다. 각 버튼에 배정된 알파벳 그룹에서, 버튼의 누름횟수에 따른 선택순서는 임의로 정해질 수 있는 것이므로 이는 선출원

에서 제시한 내용을 세부적으로 설명한 것으로 볼 수 있을 것이다. 병음음절끝자음은 음절의 처음 그리고 끝에 모두 사용될 수 있는 자음이므로 비교적 자주 사용되는 자음으로 볼 수 있고, 이렇게 모호성이 발생할 수 있는 것은 자주발생하는 경우가 아니라고 볼 수 있으며, 모호성이 발생할 수 있는 경우에도 병음/한자 색인을 참조하여 존재하는 않는 병음(예에서 "rewin")을 제외시킬 수 있으므로, 입력의 편리도 등을 고려하여, 병음음절끝자음이 속한 각 그룹에서 병음음절끝가능자음이 선택되는 순서를 정할 수 있을 것이다.

72> 이상의 내용을 포함하고 있는 도10-1에서 각 로마알파벳당 평균 1.5타에 입력할 수 있으며, 실제 중국어에서의 사용빈도를 고려하여 빈출 알파벳을 1타로 선택되도록 하면 더 적은 입력타수로 입력이 가능할 것이다.

73> 12.1.4 컨트롤버튼으로 사용되는 기능버튼의 본래 기능 입력

74> 도10-1에서 로마알파벳 모음에 첨자가 붙는 변형알파벳 입력시, 3*4 키패드 밖의 임의의 "별도의 버튼" 을 컨트롤버튼으로 활용하여 입력할 수 있다. 예를 들어 중국어 입력모드에서 좌(左)측 이동버튼([<])을 컨트롤버튼으로 활용한다면, ¯e = e + [<], ˊe = e + [<]+[<], ˇe = e + [<]+[<]+[<], ˋe = e + [<]+[<]+[<]+[<], ^e = e + [<]+[<]+[<]+[<]+[<] 와 같이 입력할 수 있으며, 좌 이동버튼을 5번 연속하여 누르면 "e" 와 결합하는 컨트롤을 더 이상 선택할 수 없으므로, 이동기능이 활성화되는 것으로 할 수 있다("e" 는 입력한 상태). 만약 삭제버튼(편의상 "[X]" 로 표시) 변형알파벳 입력을 위한 컨트롤버튼으로 활용한다면, ¯e = e + [X], ˊe = e + [X]+[X], ˇe = e + [X]+[X]+[X], ˋe = e + [X]+[X]+[X]+[X], ^e = e +

[X]+[X]+[X]+[X]+[X] 와 같이 입력할 수 있으며, 삭제 버튼을 5번 연속하여 누르면 기 입력한 "e" 가 삭제되는 것으로 할 수 있다. 이렇게 다른 기능을 가진 버튼을 컨트롤버튼으로 활용하되, 컨트롤버튼의 누름횟수에 따라 컨트롤이 선택되고 나서, 더 이상 선택되는 컨트롤이 없는 경우 본래의 기능(예. 이동기능)이 선택되는 것으로 할 수 있다.

<5> 위의 예에서 좌 이동기능이 활성화되고 나면, 단어시작 상태가 되므로 좌측 이동버튼 [<] 을 한번만 누르더라도 좌측이동기능이 입력되게 된다. 즉 "e" 를 입력후 컨트롤버튼인 [<] 버튼을 5번 누르면, "e" 만이 입력된 상태에서 커서가 좌측으로 한칸 이동하게 되며, 또 한번 누르면, 다시 옆으로 한칸 이동하게 된다. 이는 모든 언어에서 적용가능한 것이다.

<76> 12.1.5 자모음분리키패드에서의 중국어제한 반복선택방법 (3*4 키패드내에서의 변형알파벳의 입력)

<77> 여기서는, 3*4 키패드 내에서 로마알파벳 모음위에 첨자가 붙는 변형알파벳(성조부호가 붙는 경우 포함)까지 입력하도록 하는 것을 보인다.

<78> 결합운모 일람표를 보면, "i" 다음에 올 수 없는 모음은 "i" 와 "u" 인 것을

알 수 있다(바꾸어 말하면 'a' 또는 'e' 또는 'o' 가 올 수 있다). 또 "u" 다음에 올 수 없는 모음은 "u"와 "i" 인 것을 알 수 있다(바꾸어 말하면 역시 'a' 또는 'e' 또는 'o' 가 올 수 있다). 따라서 도10-2에서와 같은 모음 배치가 가능하다. 도10-2에서 좌측의 모음 "i, a, e"에 성조기호를 붙일 때는 우측의 "u" 가 배정된 [#] 버튼을 컨트롤버튼 처럼 사용하면 된다. 마찬가지로 우측의 모음 "o, u"에 성조부호를 붙일 때는 좌측의 [*] 버튼을 성조기호 컨트롤버튼처럼 사용하면 된다. 예를 들어 /o(제2성) = o + [*]+[*] = [0]+[0]+[0] + [*]+[*] 으로, /a (제2성) = a + [#]+[#] = [0] + [#]+[#] 으로, ^e = e + [#]+[#]+[#]+[#]+[#] = [0]+[0] + [#]+[#]+[#]+[#]+[#] 으로 입력할 수 있다. 복운모 ao = a + o = [0] + [0]+[0]+[0] 으로 입력할 수 있으며, 모음 a 다음에 모음 a, e 등이 올 수 없고, oa 와 같은 경우도 없으므로, 모호성 없이 시스템에서 식별할 수 있다. 모음 ··u 는 모음 u 와 모양에 있어서 유사성이 있으므로, u 가 배정된 버튼을 2타(은닉형 반복선택방법)로 입력할 수 있다.

79> 이는 중국어에서 모음 "a, e, o" 중 a 와 e 다음에 모음 u 가 나오지 않는 성질, 모음 o 다음에 모음 i 가 나오지 않는 성질을 이용한 것이다. 앞에서 설명한 제약조건, 즉 a 와 i, 그리고 e 와 i 그리고 o 와 u 는 같은 그룹으로 두지 않고, 3개의 버튼에 모음을 2개씩 배정하면서, 중국어의 로마알파벳 모음 결합 규칙을 이용하여 도10-2에서 처럼 모음이 배정된 버튼을 컨트롤버튼처럼 사용하는 것은 불가능한 것을 알 수 있다. 도10-2에서 i 와 u 를 [*] 버튼 또는 [#] 버튼에 둔 것은, 이들 버튼을 컨트롤버튼처럼 사용하는데 인식을 용이하게하기 위한 점 그리고 배치의 균형성 등을 고려한 것이다. 도10-2에서의 모음의 평균 입력타수 역시 도10-1에서와 마찬가지로 약 1.5 타이다.

380> 도10-2의 변형사례로 [0] 버튼에 배정된 3개의 알파벳 중 일부를 4*3 키패드 밖의 "별도의 버튼"으로 분리할 수도 있다. 예를 들어 상/하/좌 버튼 중의 임의의 버튼으로 3개 알파벳 중 일부의 알파벳을 분리하여 배정할 수 있는 것이다.

381> 도10-1 ~ 도10-4에서 반복선택방법을 적용하면서도 모호성없이 한어병음을 입력할 수 있는 것은 자음버튼([1] ~ [9] 버튼)과 모음버튼 [*], [0], [#] 버튼)을 분리하였고, 중국어를 한어병음으로 표기할 때의 로마알파벳의 자음과 모음의 출현규칙을 이용하였기 때문이다.

382> 12.2 자모음분리키패드에서의 영어제한 반복선택방법

383> 영어 등의 경우에 있어서도 중국어의 경우와 유사하게 도10-1 ~ 도10-4와 유사한 자모음분리키패드(CVSK)에서 반복선택방법을 적용하여 적은 모호성으로 입력할 수 있다. 이는 로마알파벳을 사용하는 모든 언어에서, 단어의 구조가 자음과 모음이 번갈아 결합하는 구조로 이루어지기 때문이다.

384> 영어의 경우 "단어시작"에서 부터 자음군이 최대한 반복되는 경우는 "strengths" 에서 와 같이 "CCCVCCCC" 의 경우라고 한다(C는 자음, V는 모음). 단어시작시 최대 3개의 자음이 올 수 있으나, 이는 "st~" 혹은 "sp~"로 시작하는 경우뿐 이라고 한다(예를 들어, spree, spleen, strength 등).

385> 도10-1와 유사하게 자음이 약 2 ~ 3개 씩 배정된 키패드에서(편의상 도10-1에서 "..u" 만이 배정되어 있지 않은 것으로 가정함) 사용자가 영어단어 "student"를 입력하기 위하여 "622~"를 입력시, 시스템은 "단어시작"후 입력된 "622"가 "sdd"를 입력한 것이 아니고 "st"를 입력한 것으로 처리할 수 있는 것이다(영어의 특정 단어에서 3개의 자음이 연속하여 나올 수 있

는 경우는 "st~" 혹은 "sp~"로 시작하는 경우라고 하였고 시스템이 이러한 영어단어생성규칙 혹은 알파벳결합규칙을 기억하고 있으면 되므로). "stu~"에서 "u" 입력시는 모음이 배정된 버튼이 눌러지므로 자음군(예에서 "st")이 끝나고 모음(예에서 "u")이 오는 것을 알 수 있다. "stu"입력 후 "d"를 입력하기 위하여 "2"가 입력되면 다시 자음이 시작되는 것을 시스템이 알 수 있다. 마찬가지로 "student"의 "~ent"를 입력하기 위하여 "~**7722" 를 사용자가 입력시, "**"이 입력되었을 때, 선출원에서 설명한 바와 같이 영어의 경우 "a"가 연속하여 등장하지 않으므로 "aa"가 아닌 "e"가 입력된 것을 시스템이 쉽게 알 수 있고, "7722"에 대하여도 "mmt", "ndd", "nt", "mmdd" 등으로 해석될 수 있으나, 선출원에서 언급한 바와 같이 "연타지연시간" 그리고 "이타지연시간"을 각기 다르게 설정함으로써, 반복선택방법에서 발생하는 모호성을 대폭 줄일 수 있다.

86> 자음과 모음이 혼재되어 각 버튼에 배정되어 있는 현재의 표준 영어키패드(도1-1)에서도 이렇게 특정언어의 단어생성규칙(알파벳결합규칙)을 반영하여 반복선택방법 적용시에 발생할 수 있는 모호성을 감소시킬 수 있다. 그러나 이렇게 자음과 모음이 하나의 버튼에 혼재하여 배정된 경우는 언어제한 입력방법(특히 반복선택방법을 풀입력방법으로 하는)을 적용하기가 매우 어렵게 된다. 예를 들어 도1-1에서 "student"를 반복선택방법을 사용하여 사용자가 입력시, "stu~" 는 "777888~" 이 되는데, "777"에 대하여 시스템에 "s" 로 인식하였다고 가정하더라도, 다음에 나오는 "888"이 "ut(즉 sut)"인지, "tu(즉 stu)"인지, "v(즉 sv)"인지 알 수 없게 된다(물론 영어제한 입력방법을 적용시 "s" 다음에 "ttt" 가 나올 수 없으므로, 시스템은 입력값 "888"에 대하여 "ttt"가 아닌 것으로 간주할 수는 있음). 마찬가지로 "~de~"를 입력하기 위하여 "333"을 사용자가 입력시에도 "de"인지 "ed"인지 알 수 없는데, "~de~"에 앞서 입력

한 "888777"이 "sut", "stu", "sv"로 해석될 수 있으므로, 3가지 경우 각각에 대하여 "~de~"와 "~ed~"의 모든 경우가 올 수 있다.

<87> 이렇게 언어제한 입력방법을 적용하더라도, 풀 입력방법으로 반복선택방법을 적용시 많은 모호성이 발생하는 이유는, 언어제한 입력방법에 있어서 자음과 모음의 출현규칙(즉, 단어 생성규칙, 알파벳결합규칙)을 이용하는데, 입력값이 자음을 입력하기 위한 것인지, 혹은 모음을 입력하기 위한 것인지가 명확히 구분되지 않기 때문이다. 도10-1 도10-4에서와 같이 자음과 모음이 각기 분리되어 배정된 경우는 반복선택방법 적용시 모음이 배정된 버튼이 눌러지면, 모음을 입력하기 위한 것으로, 자음이 배정된 버튼이 눌러지면 자음을 입력하기 위한 것으로 시스템이 인식할 수 있기 때문에 대폭 모호성을 줄일 수 있게 되는 것이다.

<88> 로마알파벳을 사용하는 영어(English)의 경우 모음이 연속하여 2개 이상 나오는 경우는 많이 있다(예. captain 에서 'ai'). 또한 실제 단어에서 같은 모음(기본모음 a, e, i, o, u)이 연속하여 나오는 경우는 food, teen 과 같이 oo,

ee 의 경우는 비교적 빈번하고, vacuum 와 같이 uu 의 경우는 드물게 있다. 출원인은 아직까지 영어사전에 존재하는 단어 중에서 ... aa ... 그리고 ... ii ... 가 연속하여 나오는 경우를 보지 못하였다. 따라서 영어의 5개 기본모음을 [*], [0], [#] 버튼에 배정함에 있어서, 'a' 를 1타로 선택되도록 하고 e, o, u 중 나머지를 2타로 선택되도록 하면 실제로 많은 경우에 모호성이 제거된다. 예를 들어 [*] 버튼에 모음 'a, o' 를 두고 반복선택방법을 적용하면, [*] 버튼이 2타 입력되었을 때, 두개의 모음 aa 가 입력된 것이 아니고 모음 o 가 입력된 것으로 시스템이 간주할 수 있게된다. 마찬가지로 [0] 버튼에 'i, u' 를 배정하고 반복선택방법을 적용하면, [0] 버튼이 2타 입력되었을 때, 역시 2개의 모음 ii 가 입력된 것이 아니고, 모음 u 가 입력된 것으로 시스템이 인식할 수 있다. 단어에서 aa, ii 와 같이 연속하여 나오는 경우가 없는 것이 절대적인 것이 아니라고 하더라도, 선출원에서 언급한 "연타지연시간" 및 "이타지연시간" 을 다르게 설정하는 것으로 모호성을 거의 없앨 수 있다. 나머지 모음 'e' 는 나머지 [#] 버튼에 배정하고 1타로 입력하도록 할 수 있다.

89> 만약 [*], [0], [#] 버튼 중 하나의 버튼을 특수한 용도의 버튼으로 사용하고 싶으면, 모음 'e' 를 모음이 배정된 임의의 버튼에 두면 된다. 예를 들어 'i, e, u' 를 [0] 버튼에 둘 수 있다. 모음 u 는 [0] 버튼 3타로 입력하게되는데, 이유는 모음 u 가 영어에서 가장 사용빈도가 낮은 모음이기 때문이다. 결국 'i, e, u' 가 [0] 버튼에 배정된 상태에서 영어에서 i 가 연속하여(즉 ...ii...) 나오는 경우가 없으면, i 와 e 는 모호성 없이 입력될 수 있고(각각 1타, 2타), u 를 [0] 버튼 3타로 입력시, u 인지, ie 인지, ei 인지 알 수 없는 모호성이 발생하게 된다. u 의 사용빈도가 낮은 것은 이러한 경우가 적게 일어나게 됨을 의미한다.

> 결국 자모음분리캐패드를 이용하여 모호성을 대폭 줄일 수 있고, 더 나아가 다수의 모음을 하나의 버튼에 배정함에 있어서, 사전에 존재하는 영어의 단어중에서 동일한 모음이 연속하여 나오지 않는 모음을 1타로 선택하도록 하여 복수개의 모음이 배정된 버튼에서 반복선택방법으로 모음을 입력함에 있어서, 거의 모호성없이 입력할 수 있게 된다.

> 이렇게 연속하여 등장하지 않는 모음을 1타로 선택하도록 하는 것, 그리고 가장 사용빈도가 낮은 모음을 3타로 선택하도록 하는 것은 버튼당 약 2~3개의 자음이 배정되는 경우에도 동일하게 적용할 수 있다.

> 위에서는 로마알파벳을 사용하는 영어의 경우를 예로 들어 설명하였으나, 로마알파벳을 사용하는 타 언어의 경우에도 이를 동일하게 적용할 수 있다.

> 12.3 자모음분리키패드에서의 인도네시아어제한 반복선택방법

> 인도네시아어도 로마알파벳(영어알파벳)을 이용하여 단어를 표기한다. 인도네시아어의 음절은 다음의 경우와 같이 구성된다고 한다. (C 는 자음, V 는 모음)

> V : be-a 관세

> VC : am-bil 잡다

> CV : go-sok 문지르다

> CVC : pon-dol 오두막

> CCV : tra-di-si 전통

> CCVC : con-trak 계약

> CVCC : teks-tur 직물

)2> CCCV : kon-struk-si 건설

)3> CCCVC : strip-tis 나체춤

)4> 자음이 3개 이상 나오는 경우는 쉽게 알 수 있듯이 영어 등에서 온 외래어이다. 이상에서 알 수 있듯이 인도네시아어의 경우도 "st~" 혹은 "sp~" 로 시작하지 않으면, 단어의 첫머리에 자음이 3개 이상 반복하여 나오지 않는 것을 짐작할 수 있다. 따라서 이러한 단어생성규칙(알파벳 결합규칙)을 이용하여 인도네시아어제한 반복선택방법을 적용할 수 있다.

05> q, x 는 물리학, 수학 등의 자연과학 기호에 사용되고 있다고 한다. 즉 문자입력에는 거의 사용되지 않는 것으로 생각된다. 따라서 q, x 는 특정 그룹에 명시적으로 배치하지 않을 수도 있고, 입력에 있어서도 3타에 입력하도록 할 수 있다. 또한 중국어의 경우와 마찬자지로 2개 이상의 로마알파벳을 조합하여 하나의 인도네시아어 음을 표기하는 경우가 있으며, ny, sy, kh, ng 등 4가지 이다.

06> 이외에 분철되지 않는 자음군으로는 bl, br, dr, dw, dy, fl, fr, gl, kr, ks, kw, pl, ps, rps, rs, sk, skr, sl, sp, spr, sr, str, sw 등이 있다고 한다. 특히 "skr~", "spr~", "str~" 등은 단어의 첫머리에 나올 수 있으므로 후술하는 언어제한 병행입력방법에서 입력값이 풀코드인지 심플코드인지를 판별하는 수단으로 활용할 수 있다. (출원인으로써, "rps" 가 단어의 첫머리에 나오는 경우가 있는지 모르나, 있다면 이 역시 마찬가지로 활용할 수 있다.)

07> 영어 알파벳 자음21개중 q, x 를 제외한 19개의 자음을 9개의 그룹으로 그룹핑하는 어떤 방법이든 가능하나, 인도네시아어의 특성을 반영하여 그룹핑하면 된다. 예를 들어,

08> BP / DT / GK / CJ / MN / LR / SZ / FV / HWY

09> 와 같이 그룹핑할 수 있다.

10> q와 x 는 적절한 그룹에 그룹핑할 수 있다. 예를 들어 q 를 "GK" 그룹에 두고, x 를 "SZ" 그룹에 그룹핑할 수 있다.

11> 인도네시아어 모음을 표기하는데에는 a, i, u, e, o 5개의 모음이 사용된다. 또 ai, au, oi 3개의 이중모음이 있으며, oi 는 매우 드물게 쓰인다고 한다. 따라서 5개의 모음을 2개 혹은 3개의 그룹으로 그룹핑하는데 있어서도 가능한한 a 와 i 가, 그리고 a 와 u 가 같은 그룹에 그룹핑되지 않도록 하는 것이 바람직할 것이다. 예를 들어, ae / uo / i 와 같이 그룹핑하는 것이다. 또한 영어의 경우와 같이 같은 모음이 연속하여 나오지 않는(연속하여 나오더라도 출현빈도가 적은) 모음을 각 그룹에서 그 모음이 속한 버튼 1타로 선택하도록 하는 것이 바람직할 것이다.

12> 12.4 자모음분리키패드에서의 일본어제한 반복선택방법

13> 일본어의 입력에 있어서 로마알파벳을 이용하여 일본어 발음을 입력하고 이를 다시 일본어로 변환하는 방법이 널리 사용되고 있었다. 따라서 도10-1 ~ 10-4와 유사한 자모음분리키패드에서 로마알파벳을 이용하여 일본어음을 입력하고 이를 일본어로 변환하는 것이 가능할 것이다. あ, い, う, え, お 는 각각 a, i, u, e, o 로 표기할 수 있다. な, に, ぬ, ね, の 는 각각 na, ni, nu, ne, no 로 표기할 수 있으며, 나머지 알파벳도 로마알파벳 자음과 모음을 조합한 형태로 표기할 수 있다.

일본어에서 로마알파벳 자음이 연속하여 2번 나오는 것은 촉음 혹은 요음(작은 글씨로 표기됨)이 사용되는 경우이다. 또한 로마알파벳 모음이 연속하여 나오는 경우는 단어중에 あ, い, う, え, お 의 조합이 연속하여 나오는 경우가 흔하지 않다는 가정하에 살펴보면, 많아야 2번 정도이며, 3번 이상 연속하여 나는 경우가 매우 드문 것을 알 수 있다. 따라서 로마알파벳의 5개의 모음(a, i, u, e, o)을 도10-1 ~ 10-4와 유사하게 3개의 그룹으로 나누어 3개의 버튼에 배정하고 반복선택방법에 의하여 모음을 입력하더라도 모호성이 별로 발생하지 않는 것을 알 수 있다. 특히 일본어의 경우 "a " 가 입력되면 대응되는 일본어가 "あ" 인 것을, "na " 가 입력되면 그에 대응되는 일본어가 "な" 인 것을 쉽게 알 수 있다(나머지 일본어 알파벳에 대해서도 동일함). 따라서 사용자가 "na" 를 입력하고, 입력값이 "na" 인 것을 시스템에서 확인하는 순간, 시스템이 사용자에게 "な" 를 제공할 수 있다.

50음도표의 일본어 표기에 사용되는 로마알파벳 자음은, k, s, t, n, h, m, y, r, w, g, z, d, b, p 의 14개 이다. 요음의 표기에는 로마알파벳 자음2개를 조합하여 표현하거나(예. cha, sha), "y" 를 조합하여 표현(예. kya)하거나, "j" 를 사용하여 표현한다. 로마알파벳 자음이 연속하여 2번이상 나오는 경우는 ch, sh, ky, ny, hy, my, ry, gy, py, py 등의 경우와 , 촉음이 사용되는 경우이다. 촉음이 사용되면, k, s, t, p 중에서 같은 알파벳이 연속하여 나오는 경우가 있게 된다고 한다(예. ippai). 따라서 16개(14개 + c, j)의 로마알파벳 자음이 일본어 입력에 필수적임을 알 수 있고, 이렇게 일본어 입력에 필수적인 자음을 입력하기 용이하도록 자모음분리키패드를 구성할 수 있다. 나머지 5개의 로마알파벳 자음(f, l, q, v, x) 역시 영어 등을 입력하기 위하여 필요할 것이나, 16개 알파벳을 중심으로 그룹핑할 수 있다. 예를 들면 다음과 같다.

> BP / DT / GK / CJ / H / MN / R / SZ / YW / => 9개 그룹으로 그룹핑

> BP / DT / GK / CJ / H / MN / Y / SZ / RW / => 9개 그룹으로 그룹핑

> BP / DT / GK / CJ / HR / MN / SZ / YW / => 8개 그룹으로 그룹핑

> 영어의 입력에 필요한 5개의 로마알파벳 자음은 중국어의 경우에서와 같이 각 그룹에 적절히 추가할 수 있다. 8개의 그룹으로 그룹핑 한 사례에서는 3*4 키패드에서 나머지 4개 버튼을 모음버튼으로 활용할 수도 있고, 모음버튼을 3개만 사용한다면, 나머지 한 개의 버튼을 영어입력에 필요한 자음을 위한 용도로 활용할 수도 있다.

> 12.5 의도적 언어제한해제

> 언어제한 반복선택방법은 사전(dictionary)에 존재하는지 여부와 상관없이 모든 단어를 입력할 수 있도록 하는 "풀입력방법"의 장점을 희생하는 것이므로 사용자로 하여금 이러한 언어제한의 제약을 둘 것인지 두지 않을 것인지를 미리 설정할 수 있도록 하는 것이 바람직하다. 그러나, 언어제한 입력방법을 가능하게 하는 "언어제한 입력모드"에서도, 사용자가 사전에 존재하지 않고 단어생성규칙(알파벳결합규칙)에 어긋나는 단어를 입력하고자 할 경우는 알파벳 입력후 특정기능(예를 들어, 공백과 좌측진행 혹은 단어종료 등)을 입력하여 타겟알파벳을 의도적으로 확정시킨 후, 다음 알파벳을 입력함으로써 모든 알파벳 조합을 입력할 수 있다. 예를 들어, 도10-1에서 "영어제한 입력모드"에서 사용자가 "622~" 를 입력하면, 시스템은 "sdd~"가 아닌 "st~"로 인식한다고 하였는데, 만약 사용자가 "sdd~"를 입력하고 싶으면 "62"를 입력 후 공백과 좌이동 기능을 입력하고 "2"를 입력하거나 혹은 "62"를 입력 후 "단어종료기능(혹은

단어종료의 효과를 줄 수 있는 컨트롤)"을 활성화시킬 수 있는 다른 어떤 수단을 입력하고 "2"를 입력하면 된다. "62" 입력후 단어종료기능이 활성화되면 시스템은 다음에 입력되는 "2"는 "단어시작" 후 처음으로 나오므로 이를 "d"로 인식할 수 있는 것이다. 이렇게 의도적으로 단어종료기능을 입력하여 특정 언어제한 입력모드에서 언어제한을 극복하는 것을 편의상 "의도적 언어제한해제"라고 부르기로 한다. "sdd~" 의 예는 "의도적 영어제한해제"인 셈이다.

<22> 마찬가지로 도10-2에서 "중국어제한 입력모드(중국어제한 반복선택방법 적용)"에서 사용자가 한어병음으로는 존재하지 않으나, 영어에는 존재하는 모음 조합인 "ui"를 입력하고 싶으면, "u" 를 입력후 단어를 종료시킬 수 있는 수단(앞의 예에서 언급)을 다시 입력하고 나서, "i"를 입력하면 된다. 그렇지 않고 도10-2를 기준으로 "중국어제한 입력모드(중국어제한 반복선택방법 적용)"에서 "u" 입력후 "i" 가 배정된 버튼을 누르면 선출원에서 언급한 바와 같이 "u" 의 변형알파벳(예. "u"에 성조부호가 붙는 알파벳 혹은 "u" 위에 ".."이 붙는 알파벳 - "..u" 를 "u"의 변형알파벳으로 간주하여 입력하도록 할 경우)이 입력되게 될 것이다(중국어제한 반복선택방법 적용시 "i" 가 배정된 버튼을 모음 "u", "o" 다음에 입력할 경우 컨트롤버튼처럼 사용한다고 하였으므로). 결국 사용자는 중국어제한 입력모드에서도 한어병음에 존재하지 않는 조합의 단어(예를 들어 영어단어 등 모든 알파벳 조합)를 입력할 수 있는 것이다. 다시 말하면 중국어를 주로 사용하는 사용자가 중국어제한 입력모드(중국어제한 반복선택방법적용 모드)로 설정한 상태에서, 그 설정을 다시 바꾸지 않고도 중국어에 존재하지 않는 모든 알파벳 조합을 입력할 수 있는 것이다. 이는 "의도적 중국어제한해제"인 셈이다.

<23> 도4-5의 한국어의 3개 모음요소를 이용한 방법에서 낱글자로써 자음 혹은 모음을 입력시에도 동일하게 적용할 수 있다. 예를 들어 도4-5에서 낱글자로써의 자음 "ㄱ"과 모음 "ㅡ"를 입력하고자 하면, "단어시작" 상태에서 [1]을 입력하고 나서 단어종료기능을 줄 수 있는 수단

을 입력하고, 다시 단어시작 상태에서 [*]을 입력하면 된다. [1] 버튼과 [*] 버튼을 연속하여 입력하면 "그"가 되기 때문이다. 표준 키보드(영어 및 한국어의 표준키보드)에서는 우측 화살표 버튼을 누르면 "단어종료" 공백이 입력되지 않고 단어종료기능일 활성화되도록 되어 있으며, 본 발명에도 우측 화살표 버튼이 추가로 구비되는 경우 이는 동일하게 적용할 수 있다.

24> 12.6 언어제한해제 지연시간

25> "연타지연시간"과 "이타지연시간" 을 하나의 버튼에 3개 이상의 알파벳이 배정된 경우에도 적용할 수 있다고 하였다. 예를 들어 도1-1의 표준 영어키패드에서 연타지연시간을 0.1초로 설정하여 [2] 버튼이 0.1초 이내에 2번 눌러지면, B 를 입력한 것으로 시스템이 간주할 수 있도록 할 수 있다.

26> 마찬가지로 [2] 버튼이 연속 3번 눌러졌을 때(즉, [2]+[2]+[2]), 첫번째 입력값과 두번째 입력값(즉, 첫번째 [2] 버튼과 두번째 [2] 버튼) 사이의 지연시간 간격이 연타지연시간으로 설정된 시간(예. 0.1 초) 이내이고, 두번째 입력값과 세번째 입력값 사이의 지연시간 간격이 연타지연시간으로 설정된 시간(예. 0.1초) 이내이면(즉 [2] + 0.1초 이내 + [2] + 0.1초 이내 + [2]), C 를 입력한 것으로 시스템이 인식하도록 할 수 있다. 혹은 [2] 버튼이 연속 3번 눌러졌을 때(즉, [2]+[2]+[2]) 전체 입력시간이 연타지연시간의 2배(예. 0.2초) 이내이면, C 를 입력한 것으로 시스템이 인식하도록 할 수도 있다.

427> 또한 영어의 경우 예를 들어 도10-1에서 "..u" 가 없는 상태를 가정하고, 영어의 모음 "a " 가 연속하여 나오지 않는 성질, "i" 가 연속하여 나오지 않는 성질을 엄격하게 적용하면, "

NII"와 같은 약자를 입력시, 반드시 단어종료기능의 입력에 의한 "의도적 영어제한해제"를 통하여 영어제한 반복선택모드에서 "NII"를 입력할 수 있을 것이다. 그러나 "NI"를 입력후 일정시간이 지나면 단어종료기능을 굳이 입력하지 않더라도 NI 가 확정되도록 할 수 있다. 이 일정시간은 선출원에서 언급한 "이타지연시간"과 동일하게 설정할 수도 있으나, "이타지연시간"보다 더 길게 설정하는 것이 바람직 할 것이다. 예를 들어 "NI"를 입력후 2초가 경과하면 단어종료기능을 입력하지 않더라도 NI가 확정되고, 시스템은 다시 "단어시작" 상태로 되도록 하는 것이다. 이 편의상 "일시적 언어제한해제 지연시간"이라 부르며, 역시 사용자로 하여금 설정할 수 있도록 하는 것이 바람직할 것이다. 이는 모든 언어에서 동일하게 적용할 수 있음은 자명하다.

28> 각종의 지연시간을 정리하면 다음과 같다.

29> 연타지연시간 ≤ 이타지연시간 ≤ 일시적 언어제한해제 지연시간

30> 3가지 지연시간은 모두 같게 설정될 수도 있으나, 연타지연시간보다는 이타지연시간을 길게, 그리고 이타지연시간보다는 일시적 언어제한해제 지연시간을 길게 설정하는 것이 바람직할 것이다.

431> 12.7 자모음분리키패드에서의 포루투갈어제한 반복선택방법

432> 포루투갈어에서 k, w, y 는 약자 혹은 외래어에만 쓰인다고 하나, 역시 문자입력을 위하여 필요할 것이다. 포루투갈어에서 겹자음은 다음과 같다고 한다.

3> bl, cl, dl, fl, gl, pl, tl,

4> br, cr, dr, fr, gr, pr, tr, vr

5> 이외에도 gn, mn, pn, ps, pt, tm, ch, lh, nh, rr, ss 와 같이 겹자음이 있다고 한다. 따라서 포루투갈어제한입력방법을 위한 자모음분리키패드를 구성함에 있어서, 위 연속하여 나오는 위 알파벳들이 같은 그룹으로 그룹핑되지 않도록 할 필요가 있다. 예를 들어 도10-* 을 참고할 수 있다. 다만 "mn" 이 같은 그룹으로 그룹핑되어 있는데, 이는 적절히 변형하여 다른 그룹으로 둘 수도 있다. 도10-* 의 키패드가 다른 언어에도 유사하게 적용될 수 있는 것은 발음의 유사성에 근거(예. 유사한 음가를 가지는 유성음과 무성음을 같은 그룹으로 ? 예. /b p/, /d t/, /g k/ … - 그룹핑)하여 그룹핑하였고, 유사한 발음을 가지는 알파벳이 연속하여 나오는 경우는 적기 때문이다.

36> 포루투갈어 기본모음은 a, e, i, o, u 의 5개 모음이 있고, 이중 a, e, o 는 강모음 I, e 는 약모음 이라고 한다. 서로 다른 2개의 모음이 연속하여 나오는 경우는 "강모음+약모음" 의 6가지(ai, au, ei, eu, oi, ou), 그리고 "약모음+약모음" 의 1가지(ui) 라고 한다. 즉 강모음과 약모음을 각각 별도의 그룹으로 그룹핑할 수 있다 (예. 도10-6에서 "..+u" 가 없는 상태). 이러한 그룹핑 이외에도 약모음과 강모음을 같은 그룹으로 되지 않도록하여 임의 개수의 그룹으로 그룹핑하는 것도 가능하다. 예를 들어 /a / e o / u i/ 와 같이 그룹핑할 수도 있다.

137> 12.8 자모음분리키패드에서의 스페인어제한 반복선택방법

스페인어에는 영어에 없는 알파벳으로 "~+n (n 위에 ~ 가 붙은 알파벳)" 이 있다. k 와 w 는 외래어의 표기에만 쓰인다고 하나, 문자입력을 위하여 역시 필요할 것이다. 스페인어에서 겹자음은 다음과 같다고 한다.

bl, cl, dl, fl, gl, pl

br, cr, dr, fr, gr, pr, tr

즉 스페인어제한입력방법을 위한 자모음분리키패드를 구성함에 있어서, l 과 r 이 함께 배정되지 않는 것이 필요하고, 위 이중자음을 구성할 수 있는 알파벳이 같은 그룹으로 그룹핑되어 같은 버튼에 배정되지 않아야 한다. 도10-1 ~ 10-4, 및 도10-6의 자음 그룹핑을 참고할 수 있다.

스페인어 기본모음은 a, e, i, o, u 의 5개 모음이 있고, 이중 a, e, o 는 강모음 I, e 는 약모음 이라고 한다. 서로 다른 2개의 모음이 연속하여 나오는 경우는 "강모음 + 약모음" 의 6가지, "약모음+강모음" 의 6가지, 그리고 "약모음+약모음" 의 2가지(iu, ui) 라고 한다. 3중모음의 경우도 "약모음+강모음+약모음" 과 같이 조합된다고 한다. 즉 강모음과 약모음을 각각 별도의 그룹으로 그룹핑하면(예. 도10-6에서 "..+u" 가 없는 상태), "약모음+약모음" 의 경우만 모호성이 있게 된다. 이 경우 iu, ui 가 모두 가능하므로, 연타지연시간/이타지연시간을 적용하여 혹은 의도적언어제한해제 등 출원인이 제시한 방법을 이용하여 모호성을 극복할 수 있다. 동일한 모음이 연속하여 등장하는 경우, 동일한 자음이 연속하여 등장하는 경우 및 그외 모호성이 발생할 수 있는 경우도 마찬가지이다.

12.9 자모음분리키패드에서의 이탈리아어제한 반복선택방법

<4> 이탈리아어에서 j, k, w, x, y 는 고어 혹은 외래어의 표기에만 쓰인다고 한다. 영어 등의 입력을 위해서는 필요할 것이다. 자음의 그룹핑에 있어서, 이 5개의 자음을 제외한 나머지 자음을 중심으로 그룹핑하고, 나머지 5개의 자음을 적절히 그룹핑할 수 있을 것이다.

<5> 이탈리아어에서 이중모음은 ia, io, ie, iu, ai, ei, oi, ui, uo, ou, eu 등이 있다고 한다. 역시 iu, ui 를 제외하면 강모음(a, e, o)와 약모음(u, I)의 조합이다. 삼중모음의 경우도 "약모음+강모음+약모음" 의 구조이다. 따라서 /a e o/, /u I/ 와 같이 강모음과 약모음의 그룹핑을 이용할 수 있으며, ui, iu 의 경우는 모호성이 있을 수 있게 된다.

<6> 12.10 자모음분리키패드에서의 독일어제한 반복선택방법

<7> 독일어의 복자음은 ch, chs, ck, ds, dt, ng, nk, pf, ph, sch, sp, st, th, ts, tz, tsch 등이 있다고 한다. 역시 연속하여 나오는 자음이 같은 그룹으로 그룹핑되지 않도록 그룹핑함에 도10-* 의 자음 그룹핑을 참고할 수 있다. 또한 독일어에서 "sch~" 의 경우는 어두에 자음이 3개 이상 나올 수 있는 경우라고 하므로 후술하는 언어제한병행입력방법에서 이를 활용할 수 있다. Tsch의 경우도 어두에 나올 수 있으나 매우 드물게 쓰인다고 한다.

<8> 독일어에는 5개의 단모음이 있고, a, o, u 위에 ".." 이 붙는 변모음이 있다. 이중모음으로는 au, ei, ai, eu, "..au" , ie, 등이 있다고 한다. 단모음(기본모음)만을 이용하여 모음을 입력시, ie, ei 의 경우가 있으므로, i 와 e 를 다른 그룹으로 그룹핑하는 것이 필요하다. 예를 들어, /a e o/, /I u/ 와 같이 그룹핑할 수 있으며, 많은 변형이 가능하다.

<9> 동일한 자음 혹은 모음이 연속하여 나오는 경우는 출원인이 제시한 여러가지 방법에 의하여 모호성을 회피할 수 있다.

0> 12.11 베트남어제한 입력방법

1> 12.11.1 자모음분리키패드에서의 베트남어제한 입력방법

2> 베트남어의 음절은 주로 "모음", "자음+모음", "모음+자음", 및 "자음+모음+자음" 의 형태로 이루어진다고 한다. 특히 베트남어에는 단음절어가 기본을 이루고 있다고 한다. 복음절어가 증가하고 있기는 하지만 단음절어가 많다는 것은 자모음 분리키패드에서 입력시스템을 구성하기 용이하다는 것을 의미한다.

53> 베트남어에는 a, e, i(y), o, u 5개의 기본모음과 "v+a (a 위에 v가 붙은 모음. 이한 'x+모음' 은 모음 위에 'x' 가 붙은 모음)", "^+a", "^+o", ",+o", ",+u" 등 6개의 변모음이 있다고 한다. "y" 는 "i" 를 길게 발음한다고 한다. 5개의 기본모음만으로 자모음 분리키패드를 구성하고 나머지 변모음을 컨트롤처리방법에 의하여 입력할 수 있다. 기본모음과 변모음 일부만으로 자모음분리키패드를 구성할 수도 있고, 혹은 11개의 모음 모두로 자모음분리키패드를 구성할 수도 있다.

54> 베트남어2중모음 및 3중모음은 그 결합형태가 다양하여 5개의 기본모음을 도10-*에서와 유사하게 2~3개의 그룹으로 그룹핑하면서 반복선택방법 적용시 모호성이 없도록 그룹핑하는 것이 쉽지 않다. 또한 베트남어에는 6가지 성조가 있으며, 5가지 성조부호가 모음 위 혹은 아래에 표시된다. 그리고 베트남어에서 동일한 모음이 연속하여 나오는 경우는 없는데, 이는 베트남어의 모음을 입력하는데 있어서 유용한 성질이다. 따라서 5개의 기본모음을 5개의 그룹으로 두고, 은닉형 반복선택방법 의하여 변모음을 입력하고, 또 컨트롤처리방법에 의하여 기본모음 및 변모음에 성조부호를 붙이는 것을 생각할 수 있다. 여기서 y 는 i 의 변모음으로 간주할

수 있다. 즉 모음을 / 'a' , 'v+a' , '^+a' / 'o' , '^+o' , ',+o' / 'u' , ',+u' / 'i' , 'y' / 'e' , '^+e' / 의 5개 그룹으로 나누는 것이다. 이를 변형하여 변모음의 수가 적은 그룹을 합하여 4개 그룹으로 만드는 것도 가능하다. 예를 들어 / 'a' , 'v+a' , '^+a' / 'o' , '^+o' , ',+o' / 'u' , ',+u' / 'i' , 'y' , 'e' , '^+e' / 의 4개 그룹으로 나누는 것이다. 단 5개의 그룹으로 두면 반복선택방법을 적용하여 모음을 모호성 없이 입력할 수 있게 된다. 각 그룹에서 임의의 알파벳을 대표알파벳으로 하여 키패드상에 표시하고 나머지 알파벳은 표시하지 않을 수 있다. 주로 단모음이 대표알파벳이 되는 것이 자연스러울 것이며, 4개의 그룹으로 나누는 사례에서, i, e 가 모두 표시되는 것도 가능하다. 버튼누름횟수에 따른 선택순서는 사용빈도 등을 고려하여 정해질 수 있다.

55> 베트남어의 자음으로는 영어에 없는 "-+d ('d' 가운데에 '- ')" 가 있고, 영어의 f, z 등이 쓰이지 않는다고 한다. '-+d' 는 'd' 의 변형알파벳으로 간주하여, (은닉형) 반복선택방법 혹은 컨트롤처리방법 등을 적용할 수 있다. 교과서에 따라서는 베트남어에서 f, w, z가 사용되지 않는다고 하기도 하고, 또 어떤 교과서에서는 반자음으로 w, j 가 사용된다고도 한다. 최소한 f, z는 사용되지 않고, w 는 드물게 사용되는 것으로 볼 수 있다. 본 발명에서는 w를 사용되지 않는 것으로 간주하나, 필요한 경우 적절한 그룹에 포함시키면 된다.

156> 또한 복자음으로 ch, gh, gi, kh, ng, ngh, nh, ph, qu, th, tr 등이 있다고 한다. 이 중 gi, qu 는 형태상으로 자음과 모음의 결합이므로 여기서는 고려하지 않는다. 베트남어가 주로 단음절어로 이루어져 있다는 것은 타 언어와 달리 "자+모+자" 로 이루어진 음절이 연속하여 나오는 경우가 거의 없다는 것을 의미하고, 위의 복자음의 경우를 제외하고는 자음과 자음이 연속하여 나오는 경우가 거의 없다는 것을 의미한다. 결국 자음을 몇 개의 그룹으로 나

누어 반복선택방법을 적용함에 있어서 위의 복자음을 구성하는 자음이 같은 그룹으로 그룹핑되지 않도록 그룹핑하면 중국어의 사례에서와 같이 자음을 모호성없이 입력할 수 있게 된다.

7> 예를 들어, b p / d t / g k / c q / s x / m n / l r / h v j와 같이 8개의 그룹으로 그룹핑할 수 있다. 또한 b p v / d t / g k q / s x c / m n j / h l r / 와 같이 6개의 그룹으로 나눌 수도 있다. 임의의 그룹으로 그룹핑하는 것도 가능할 것이다. f, z, 등의 사용되지 않는 자음은 적절한 그룹에 추가적으로 포함시킬 수 있다.

8> 자음을 8개의 그룹으로 나누고, 모음을 4개의 그룹으로 나누면, 3*4 키패드내에서 자음과 모음을 모두 수용할 수 있다. 자음을 6개의 그룹으로 나누고, 모음을 5개의 그룹으로 두면 11개 버튼에서 자음과 모음을 수용할 수 있으며, 나머지 하나의 버튼을 모음에 성조부호를 붙이기 위한 컨트롤버튼으로 사용할 수 있다. 자음을 6개 그룹으로 나누고, 모음을 4개 그룹으로 나누면, 10개의 숫자버튼으로 자음버튼과 모음버튼을 모두 수용할 수 있으며, 이는 심플코드 활용에 있어서 심플코드가 숫자로만 이루어지게 하는 효과가 있다. 나머지 2개의 버튼은 성조부호를 입력하기 위한 컨트롤버튼 및 모호성을 제거하기 위한 후속컨트롤버튼으로 활용할 수 있다.

59> 12.11.2 자음과 모음의 쌍을 이용하는 베트남어제한 입력방법

60> 베트남어의 모음과 자음을 쌍으로 그룹핑하고 반복선택방법을 적용할 수 있다. 한국어의 경우를 참고할 수 있다. 자음과 모음의 쌍을 10개로 하면, 나머지 자음과 모음은 컨트롤처리방법에 의하여 입력하면 된다.

1> 12.12 자모음분리키패드에서의 러시아어제한 입력방법

2> 러시아어를 위한 자모음분리키패드를 구성하기 위하여 출원인의 선출원 PCT/KR02/00247을 참고할 수 있다.

3> 러시아어에는 33개의 알파벳이 존재한다. 그중 10개는 모음이고, 20개는 자음이며, 1개의 반모음(혹은 반자음), 그리고 2개의 기호알파벳(경자음부호, 연자음부호)으로 구성되어 있다.

4> 다음은 33개의 자모를 알파벳 순서에 따라 대문자와 소문자를 나열한 것이다.

5> А Б В Г Д Е Ё Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я

6> а б в г д е ё ж з и й к л м н о п р с т у ф х ц ч ш щ ъ ы ь э ю я

7> 20개의 자음은 다음과 같이 유성자음과 무성자음으로 나누어진다. 괄호안은 발음을 의미한다.

8>

무성자음	п, т, к, ф, с, ш (p), (t), (k), (f), (s), (sh)	х, ц, ч, щ (x), (ts), (tsh), (shsh)	
유성자음	б, д, г, в, з, ж (b), (d), (g), (v), (z), (zh)		л, р, м, н (l), (r), (m), (n)

9> 모음은 경모음과 연모음으로 나누어진다.

10>

경모음	а(a), э(e), ы(y), о(o), у(u)
연모음	я(ja), е(je), и(i), ё(jo), ю(ju)

11> 러시아어의 자음을 적절하게 그룹핑하는데 있어서, 유사한 음가를 가지는 유성음과 무성음을 그룹으로 둘 수 있다. 예를 들어 /б(b) п(p) / д(d) т(t) / … 와 같이 분류하는 것

이다. 발음상 쌍을 이루지 않는 자음들은 선출원에서 제시한 바와 같이 적절히 그룹핑할 수 있다. 몇가지 예를 들면 다음과 같으며, 쌍을 이루는 무성음과 유성음을 함께 그룹핑하면서 다양한 변형이 가능하다.

2> 10개 그룹핑

3> 사례1 : Б П/ Д Т/ Г К/ В Ф/ З С/ Ж Ш/ Л Р/ М Н/ Х Ц/ Ч Щ

4> 사례2 : Б В/ Г К/ Д Т/ Ж З/ Л Р/ М Н/ П Ф/ С Х/ Ц Ч/ Ш Щ

5> 9개 그룹핑 사례 : (b) п(p) / д(d) т(t) /г(g) к(k) / в(v) ф(f) х(x) / з(z) с(s) / ж(zh) ш(sh) / ц(ts) ч(tsh) щ(shsh) / л(l) р(r) /м(m) н(n)

76> 8개 그룹핑 사례 : (b) п(p) / д(d) т(t) /г(g) к(k) / в(v) ф(f) х(x) / з(z) с(s) ж(zh) / ш(sh) ц(ts) ч(tsh) щ(shsh) / л(l) р(r) /м(m) н(n)

77> 7개 그룹핑 사례 :

78> . . . . . .

79> 위의 예에서 반자음 й 및 경자음 부호 ъ와 연자음 부호 ь를 적절한 그룹에 포함시킬 수도 있다. 또한 이러한 그룹을 각 버튼에 배정함에 있어서 키패드를 간결하게 유지하기 위하여 일부의 알파벳(예. й, ъ, ь 및 이외의 알파벳) 을 생략하고 표시하지 않을 수도 있다.

0> 모음 역시 경모음과 연모음의 쌍을 이용하여 10개의 모음을 5개의 그룹으로 그룹핑할 수 있다. 러시아어의 모음 알파벳은 10개가 있고, 발음으로는 11개의 모음 소리가 있으나, 모음의 기본음소는 (a), (e), (i), (o), (u) 의 5개가 있다고 한다. 이중 (e), (o) 는 강세 아래에서만 나타난다고 한다. 즉 모음 음소 (a), (o)를 강모음으로 볼 수 있을 것이다. 강세가 없는 위치에서의 모음음소는 나머지 3개 라고 한다. 따라서 러시아어의 10개 모음 알파벳 중 5개의 기본 알파벳을 두고, 이중 강모음에 해당하는 모음을 하나의 그룹으로 또 나머지 나머지 3개의 기본모음을 또 하나 혹은 하나 이상의 그룹 나눌 수 있을 것이다. 같은 종류의 모음이 연속하여(예. 강모음 다음에 연속하여 강모음) 나오기 어려운 것을 예상할 수 있기 때문이다. 예를 들어 a(a) o(o) / y(u) э(e) и(i) / 와 같이 2개의 그룹으로 혹은 a(a) o(o) / y(u) э(e) / и(i) / 와 같이 3개의 그룹으로 둘 수 있다. 기본모음과 대응되는 쌍을 이루는 모음은 기본모음이 속한 그룹에 함께 두고 반복선택방법을 적용하거나, 기본모음의 변형알파벳으로 간주하여 컨트롤처리방법을 적용할 수 있다.

81> 자음과 모음의 그룹을 적절하게 이용하여 3*4 키패드내에서 자모음분리키패드를 구성할 수 있다(예. 모음 5개 그룹과 자음 7개 그룹, 모음3개 그룹과 자음 9개 그룹). 러시아어에서 영어와 마찬가지로 다수의 자음이 연속되는 경우가 있으나(예. CCCVC...), 전형적인 음절구조는 자음과 모음의 반복구조(CV, CVCV, CVCVCV, …) 라고 한다. 이는 자모음 분리키패드에서 반복선택방법을 적용시 대부분의 경우 모호성 없이(즉, 매우 적은 모호성으로) 러시아어 단어를 입력할 수 있는 것을 의미한다.

182> 12.13 자모음분리키패드에서의 힌디어제한 입력방법

3> 출원인의 선출원 PCT/KR00/00601 에서 힌디어 자음을 발음의 유사성에 근거하여 9개의 그룹으로 그룹핑하는 사례를 보였고, 본 발명에서 10개의 그룹으로 그룹핑하는 사례를 보였다. 힌디어의 경우도 발음의 유사성을 강하게 가진 그룹자음을 같은 그룹으로 그룹핑할 수 있음을 보였다. 예를 들어 앞에서 제시한 그림에서 (k), (kh), (g), (gh) 의 발음을 가진 그룹자음을 하나의 그룹으로 그룹핑하는 것이다. 나머지 유사발음 그룹에 대해서도 마찬가지이다. 위에서 제시한 힌디어의 35개 자음 중 아랫점이 있는 __(ud), __(udh) 를 제외한 33개 자음을 9개 그룹 혹은 8개 그룹으로 임의로 그룹핑할 수 있다. __(ud), __(udh) 는 각각 __(d), __(dh)의 변형알파벳으로 간주하여, 기본알파벳이 속한 그룹에 둘 수 있다.

34> 다음은 선출원에서 기술한 힌디어의 모음이다.

85>

기본형 :	a अ	aa आ	i इ	ee ई	u उ	oo ऊ	ae ए	aae ऐ	o ओ	au औ	ri ऋ			
축약형 : 없음		aa ा	i ि	ee ी	u ु	oo ू	ae े	aae ै	o ो	au ौ	ri ृ			

86> 이하에서 편의상 힌디어 모음을 밑줄 옆 괄호안에 영어 발음으로(밑줄에는 해당 발음의 힌디어 알파벳이 있는 것으로 간주) 혹은 단순히 괄호안에 영어알파벳으로 표기하기로 한다. 편의상 __(aa) 는 (a-)으로, __(ee)는 (i-)로, __(oo)는 (o-)로, __(ae)는 (e)로, __(aae)는 (ai)로 표기한다.

87> 모음 __(ri)는 자음으로 분류하기도 한다고 한다. 모음 __(ri)를 제외한 10개의 모음을 위에서와 같이 2개씩 쌍을 지어 5개의 그룹으로 그룹핑할 수도 있다.

88> 힌디어의 모음은 (a), (i), (u)의 단모음과 __(a-), __(i-), __(u-)의 장모음으로 크게 나눠지며 나머지 4개의 모음은 복합모음으로 단모음의 조합으로 표시될 수 있다고 한다. 즉

__(e) = (a) + (i) or (a) + (i-), __(ai) = (a) + (e), __(o) = (a) + (u) or (a) + (u-), __(au) = (a) + (o) 와 같이 된다고 한다. 따라서 기본모음 (a), (i), (u)를 각각 3개의 그룹으로 하여 3개의 버튼에 배정하고, 각 버튼 1타로 기본모음을 입력하고, 기본모음에 대응되는 장모음은 기본모음이 배정된 버튼 2타로 입력하고, 나머지 4개의 복합모음은 기본모음의 조합으로 입력할 수 있다.

39> 예를 들어 (a), (i), (u) 가 각각 [*], [0], [#] 버튼에 배정되어 있을 경우, __(a-) = ** 로, e = *0 혹은 *00 로 입력하는 것이다. 만약 **0 가 눌러지면 "**" 까지 눌러졌을 때 시스템은 __(a-)로 인식하나 다음에 "0" 이 눌러지는 순간 **0을 (a) + (e) = __(ai)로 인식할 수 있게 된다. 마찬가지로 **# 이 눌러지면 "**" 까지 눌러졌을 때 시스템은 __(a-)로 인식하나 다음에 "#" 이 눌러지는 순간 **#을 (a) + (o) = __(au)로 인식할 수 있게 된다

.

90> 이는 힌디어의 10개 모음이 하나의 단어에서 연속하여 나오지 않는 경우에 가능하다. 힌디어의 모음이 연속하여 나오는 경우가 있다면, 모호성이 발생할 수 있을 것이다. 즉 **0 입력시, __(a-) 와 __(i) 를 입력한 것인지, __(ai) 를 입력한 것인지 알 수 없게 되는 것이다. 그러나 10개의 기본모음이 연속하여 나오는 경우가 있다고 하더라도, 그 빈도가 적으면, **0 과 같이 복합모음 __(ai) 를 의미할 수 있는 경우에 기본적으로 시스템이 입력값을 복합모음으로 인식하는 언어제한을 적용할 수 있다. 이 경우 사용자가 장모음과 __(a-) 와 단모음 __(i) 의 조합을 입력하려면, "의도적언어제한해제" 혹은 "언어제한해제지연시간" 을 이용하여 해결할 수 있다. 즉 예에서 ** 을 입력하고 일정시간만큼 혹은 의도적으로 단어종료기능을 활성화시키고 나서 0 를 입력하는 것이다.

1> 힌디어 역시 주된 음절구조가 자음과 모음이 번갈아 나오는 형태이나, 자음이 연속하여 나오는 경우도 있다. 수직선으로 끝나는 자음 다음에 또 자음이 나올 경우 수직선이 없어지면서, 다음 자음과 결합하여 결합자음을 이룬다. 수직선이 없는 자음은 알파벳 아래에 "," 비슷한 기호가 붙으면서 다음 자음과 결합하게 된다고 한다. 이외에도 몇가지 규칙이 있으며 불규칙적인 경우도 있다고 한다. 이렇게 자음이 연속하여 입력된 것으로 인식(일시적으로 인식 혹은 확정적으로 인식)되는 경우, 결합자음으로 표시하여 주는 오토마타를 구현하는 것은 용이한 일이다.

92> 영어를 모국어로 사용하지 않는 나라에서도 키패드상 영어알파벳을 병기하여 영어의 입력에 활용하고 있다. 인도에서는 힌디어와 영어를 공용어로 사용하므로, 유사발음을 가진 힌디어자음과 영어의 자음을 같은 버튼에 배정함으로써 심플코드의 활용 및 문자입력에 자연스러움을 더할 수 있다. 예를 들면 다음과 같으며 변형이 가능하다.

93>

힌디어 자음	영어 자음	힌디어 자음	영어 자음
k क kh ख g ग gh घ	GK	n ङ n ञ n ण n न m म	MN
ch च chh छ j ज jh झ	CJ	y य v व	YV
t ट tth ठ d ड dh ढ ud ड़ udh ढ़	DT	sh श shh ष s स	SZ
t' त th' थ th द thh ध	H	r र l ल	RL
p प f फ b ब bh भ	BP		

94> 위에서 z,w 등과 같은 영어자음은 중국어의 경우에서와 같이 적절한 그룹에 추가로 혹은 별도로 둘 수 있다. 또한 키패드상의 배치를 간결하게 하기 위하여 각 그룹에서 일부의 알파벳만을 키패드상에 표시할 수도 있다(은닉형반복선택방법 적용). 예를 들어 위의 그룹핑에서

첫번째 그룹을 임의의 버튼에 배정시 _(k) 와 _(g) 만을 배정할 수 있다. 각 그룹에서 버튼누름횟수에 따라 선택되는 순서는 임의로 정하면 된다. 모음 _(ri) 를 R L 이 배정되는 버튼에 배정할 수도 있다.

5> 역시 키패드상의 배치를 간결하게 하기 위하여 영어모음(영어알파벳 모음)만을 표기할 수 있다. 앞의 예에서 보였듯이 힌디어 모음의 경우는 영어 모음과 발음상 유사성을 강하게 가지므로 영어 모음만을 표기하는 것이 매우 자연스럽다. 더 나아가 자음의 경우도 위의 힌디어 자음과 영어 모음의 대비표에서 보였듯이 발음의 유사성을 강하게 가진, /G K/, /M N/, /D T/, /B P/, /R L/ . . . 의 그룹과 대응되는 힌디어 자음을 생략하여 표시하지 않고 영어 자음만을 표시함으로써, 키패드를 더욱 간결하게 할 수 있다. 물론 이렇게 힌디어 알파벳이 생략된 경우에도 경우에도, 중국어의 예에서 중국어제한 입력모드에서 병음을 모호성없이 입력할 수 있었던 것과 마찬가지로, 힌디어제한 입력모드에서는 힌디어제한에 따라 모호성없이 힌디어를 입력하게 된다.

96> 12.14 반자음분리 자모음분리키패드에서의 언어제한 입력방법

97> 각 언어별로 반자음(즉, 반모음)은 조금씩 다르다, 영어에서는 일반적으로 w, y, j 등을 반자음으로 취급하고 있으며, 어떤 언어에서는 j 를 제외하기도 하고, 또 어떤 언어에서는 v 등을 포함하기도 한다. 일반적으로 w, y 는 거의 모든 언어에서 공통적으로 반자음으로 취급하고 있는 것으로 보인다.

98> 여기서 반자음은 모음과 유사한 성질 및 음가를 가지므로, 반자음을 별도의 그룹으로 두는 것은 발음의 유사성에 근거하여 자음을 그룹핑하는 것의 연장선상에 있는 것이다.

9> 또한 언어별로 영어알파벳 중 특정 언어에서 사용하지 않는 알파벳(예를 들어 스페인어에서 외래어 등에만 사용된다고 하는 k 와 w)이 있기도 하고, 영어알파벳 이외의 알파벳(출원인은 편의상 변형알파벳 혹은 추가알파벳이라 하였음)이 쓰이기도 한다. 영어알파벳 중 사용되지 않는 알파벳 역시 보편적으로 쓰이는 영어의 입력을 위하여 필요할 것이다. 따라서 특정 언어에서 사용되지 않는 로마알파벳의 수가 많지 않으면 반자음과 함께 그룹핑하고 같은 버튼에 배정할 수 있다. 중국어에서의 도10-6을 참고할 수 있다.

)0> 또한 특정 언어에서 사용되지 않는 알파벳들을 함께 그룹핑하여 같은 버튼에 배정할 수도 있다.

01> 12.15 불완전 자모음분리키패드에서의 언어제한입력방법

02> 본 발명에서 각 언어에 최적화된 자모음분리키패드를 구성하고 반복선택방법을 적용함으로써 모호성을 제거하거나 최소화할 수 있음을 보였다. 또한 이렇게 자모음분리키패드를 구성함으로써, 입력시스템의 알고리즘을 단순화할 수 있다. 이를 근간으로 하여 다양한 변형이 가능함은 자명하다. 따라서 이를 약간 변형하여 일부의 버튼에 자모음을 함께 배정하는 경우도 본 발명에서 제시한 자모음분리키패드에서의 언어제한입력방법의 변형된 범주이다. 이를 편의상 "불완전 자모음분리키패드" 라 부르고, 자모음이 함께 배정된 버튼을 편의상 "자모음혼합버튼" 이라 부르기로 한다.

503> 예를 들어 자모음분리키패드에서 모음 "i" 와 (반)자음 "y" 를 함께 그룹핑하고, 특정 버튼에 배정하고 버튼누름횟수에 따라 소정의 순서에 의하여 자음(여기서는 반자음)과 모음이 선택되도록 할 수도 있다(나머지 버튼은 자음버튼과 모음버튼으로 분리). "iy" 혹은 "yi"

처럼 연속하여 나오는 경우가 없으면, 중국어제한입력방법을 적용하여 역시 모호성없이 한어병음입력이 가능하다. 이 경우 자모음이 혼합배정된 특정버튼의 누름횟수에 따라 "y" (한번 누름), "i" (두번 누름) 의 순서로 선택된다고 가정하고, 자음버튼이 눌러지고 나서 이 자모음혼합버튼이 한번 눌러지는 순간, 이전 자음버튼의 입력이 자음을 입력한 것으로 확정되므로, 시스템은 자모음혼합버튼이 한번 눌려진 것에 대하여 일시적으로 "y" 가 입력된 것으로 간주하지 않고, 모음 "i" 가 입력된 것으로 확정할 수 있다. 중국어에서 특정 자음 다음에 자음 "y" 가 나올 수 없는 언어제한을 이용할 수 있기 때문이다.

0404> 또 다른 예를 들어 모음 "i" 와 자음 "j" 를 혹은 모음 "i" 와 자음 "k" 를 같은 그룹으로 그룹핑하고, (언어제한)반복선택방법을 적용할 수 있다. 한어병음 입력시 "…ik…" 혹은 "…ki…" 와 같이 연속하여 나오는 경우가 있으면 모호성이 발생할 수 있다. 특별한 이유없이 이렇게 일부의 버튼에 자음과 모음을 함께 배정하는 것은, 입력알고리즘을 복잡하게 만들고 입력시스템 구성의 효율성을 떨어뜨리며, 완전한 자모음분리키패드에서의 언어제한입력방법을 적용하는 것보다 좋지 않다. 다만, 각 언어의 특성상 10개의 숫자버튼에 모두 자음을 배정하기 위한 특별한 목적 등을 위하여 일부의 버튼에 자모음혼합버튼을 둘 수도 있다 (한국어의 사례 참조).

0505> 이는 도10-7의 흐름도를 변형하여 적용할 수 있다. 도10-9를 참고한다. 자모음혼합버튼이 한 개 이상 있는 경우도 동일하며, 하나의 자모음혼합버튼에 2개 이상의 자모음이 배정된 경우도 동일하다. 불완전 자모음분리키패드에서의 언어제한을 적용하는 것은 조금 더 복잡하며 (위의 "i" 와 "y" 의 사례 참조), 각 언어별 자음과 모음의 결합규칙을 적용하여야 하므로, 각 언어별 언어제한을 세부적으로 적용한 흐름도는 도10-9보다 더 복잡해질 수 있다. 도10-9는 도10-7을 일부 변형하고 일반적인 내용만을 매우 간략히 표현한 것으로 생각하면 된다.

6> 예를 들어, 도5-1의 표준 영어키패드에서 중국어 병음 입력시, 5개의 모음이 각기 다른 버튼에 배정되어 있으므로 모음은 모호성 없이 인식될 수 있다. 자음의 경우 "~모음+n" 다음에 자음이 오는 경우는 모호성없이 인식될 수 있다. "~n+X" ('X'는 성모를 영어자음으로 표시한 것) 에서 "~모음+n+m"이 입력될 경우, 모음이 [6] 버튼에 함께 배정된 모음 "o"가 결합될 수 있는 모음(예를 들어 모음 'a') 이 오면 일시적으로 "~ao" 로 인식될 수 있다. 그러나 정상적인 중국어 병음음절이 "m"으로 끝나지 않으므로, 다음 입력값에 의하여 인식되는 모음을 보아 그 이전 입력값을 "~anm+모음"으로 인식할 수 있다. 만약 "a" 가 결합할 수 없는 모음(예를 들어 'e') 다음에 "n+m = 666"이 입력된 경우 "eo"의 결합이 불가능한 언어제한을 이용하여 "~eo" 가 아닌, "~enm~"으로 처리할 수 있다 (중국어 병음에서 '~emn~'은 불가능 하므로).

07> "~모음+ng"로 끝나는 음절 다음에 자음이 올 때, 모호성이 발생할 수 있다. 예를 들어, "~ngg~" 가 "~nh~"로 인식될 수 있는 것이다. 중국어의 경우 시스템은 "~ngg~" 및 "~nh~"와 일치하는 한자를 모두 색인에서 검색하여 사용자에게 제공할 수 있다. "~ngh+모음~" 의 경우는 "~ni+모음" 로 인식될 수 있다. 만약 뒤이어 인식되는 모음이 "i"다음에 결합할 수 없는 모음(예를 들어 'u') 이면 입력값을 "~nghu"로 인식할 수 있으나 그렇지 않은 경우는 마찬가지로 "~ngh+모음~" 및 "~ni+모음~" 모두에 대응하는 한자를 검색하여 처리할 수 있다. 이외에도 일부 모호성이 발생하는 경우에 가능한 해석결과와 일치하는 단어를 모두 검색하여 사용자에게 제공할 수 있다.

508> 12.14 비은닉형 컨트롤처리방법의 적용

이상의 자모음분리키패드에서 임의의 버튼(3*4키패드 내의 버튼 혹은 외부의 별도의 버튼)을 컨트롤버튼으로 정하고, 대표알파벳만을 해당 버튼 1타로 선택되도록 하고, 나머지 알파벳을 컨트롤처리방법에 의하여 입력할 수 있다. 예를 들어 도10-6의 키패드에서 [0] 버튼에 배정된 "w, y, v" 를 임의의 자음버튼에 배정하고, [0] 버튼을 컨트롤버튼으로 활용하여, "p = b+0 = 10" 로 입력할 수 있는 것이다 (컨트롤 후입력적용시). 이는 이미 선출원에서 언급한 내용이다.

13. 입력값의 길이로 판단하는 병행입력방법

색인에 저장된 단축코드의 길이는 경우에 따라 특정 개수 이하로 될 수 있다. 또한 특정한 경우 사용자에 의하여 혹은 시스템에 의하여 입력되는 단축코드의 유형(예를 들어, 유형1(도시명), 유형2(은행명), . . . 선출원에서 유형1, 유형2는 트리형태로 그룹지어질 수 있다고 하였다)이 지정될 수 있다. 만약 유형1(도시명)으로 입력되는 단축코드의 유형이 지정되고, 도시명의 단축코드의 길이가 3 이하라고 가정하면, 단축입력방법을 기본모드로 하는 병행입력방법 적용시, 입력값의 길이가 3을 초과하는 순간, 시스템이 입력값을 풀코드로 간주하여 처리할 수 있는 것이다. 반대로 풀우선 병행입력방법 적용시, 3개의 입력값이 입력되고 단어종료기능(예. 공백)이 입력되면, 시스템이 입력값을 심플코드로 간주하여 처리할 수 있다.

이는 한국어, 중국어와 같이 특정한 유형(도시명 ? 北京, 회사명 ? 四通集團, . . .)의 단축코드의 길이가 일정 길이 이하로 될 수 있는 경우에 특히 유용할 것이다(한국어 중국어

등에 있어서 특정 유형의 명사는 일정 수 이하의 음절로 이루어진 경우가 많고, 음절기준 이니셜코드 단축코드로 이용하는 것이 자연스러우므로. 예를 들어 한국어서 대부분의 은행명은 2 음절로 이루어져 있으므로, 은행명에 대한 음절기준 이니셜 코드의 길이는 2가 됨).

3> 이는 입력값이 단축코드인지 풀코드인지를 입력 초기에 판단하여 사용자에게 제공하지 않아도 되는 경우, 즉 입력값의 길이가 특정 길이 이상인지, 이하인지만으로 입력값이 심플코드인지 풀코드인지 판단하여도 되는 경우에 유용할 수 있다. 즉 매번 입력값이 입력될 때마다 단축코드 색인을 검색하지 않고, 단지 입력값의 길이가 특정 길이 이상인지만을 체크하면 되므로 구현을 단순화하고 시스템의 성능을 향상시킬 수 있는 것이다.

14> 14. 언어제한 병행입력방법

15> 언어제한 병행입력방법은 풀우선 병행입력방법 적용시, 입력값이 특정언어의 "단어생성규칙" 을 위배하는 순간(즉, 언어제한을 위배하는 순간), 시스템이 입력값을 심플코드로 간주하여 처리하는 것이다. 만약 입력값과 일치하는 값이 심플코드 색인에서 더 이상 존재하지 않는 것이 확인되면, 다시 입력값을 풀코드로 간주하여 처리할 수 있다.

16> 14.1 언어제한 반복선택방법을 풀입력방법으로 하는 언어제한 병행입력방법

17> 14.1.1 중국어

8> 특히 도10-1 ~ 10-4를 기준으로한 중국어의 음절기준 이니셜코드는 자음버튼인 [1] ~ [9] 의 숫자값만을 가지게 된다(모음 a, o, e 가 단독으로 사용되는 경우도 있다고 하나, 감탄사 정도이며 현실적으로 거의 없다고 하고, 모음 i, u, ...u 이 단독으로 사용될 경우는 앞에 y, w, y 를 각각 붙이므로). 반복선택방법을 적용한 풀코드는 도10-1 ~ 10-4의 예에서 대부분의 경우 2번째 혹은 적어도 3번째 입력값은 모음 입력을 위하여 [*], [0] 혹은 [#] 버튼 중의 한개 버튼이 입력되어야 한다. 이러한 성질은 선출원의 한국어의 예(모음요소를 활용한 방법)에서 언급한 바와 같이 "병행입력방법"을 적용시 입력초기에 입력값이 풀코드를 이루는지, 아닌지를 알 수 있는 좋은 특성을 제공한다.

19> 예를 들어 풀입력방법으로 도10-1 ~ 10-4를 기준으로한 반복선택방법을 사용하고, 풀입력모드를 기본입력모드로 하는 병행입력방법(풀우선 병행입력방법)에서, 입력값이 "693 ... = shk ..." 가 입력되면, 3번째 값 [3] 이 입력되는 순간 입력값이 풀코드를 이루지 못하는 것('sh' 까지는 가능해도 그 다음 자음 'k' 가 올 수는 없으므로)을 시스템이 인지할 수 있으므로, 시스템이 입력값을 심플코드로 간주하여 처리할 수 있다. 역시 "112 … = pd …" 가 입력되면, 3번째 입력값 [2] 가 입력되는 순간 입력값이 풀코드를 이루지 못하는 것(중국어 입력에서 '11' 이 'bb' 가 될 수 없고 'p' 로 해석될 수 밖에 없으며, p 다음에 로마알파벳 자음 d 혹은 t가 오면 중국어 음절을 이룰 수 없으므로)을 시스템이 인식하고, 시스템이 입력값을 심플코드로 간주하여 처리할 수 있다. 마찬가지로 "7771 … = wb…" 이 입력되면 4번째 입력값 [1] 이 입력되는 순간 시스템이 입력값을 심플코드로 간주하여 처리할 수 있다. 결국 대부분의 경우 2번째 혹은 3번째 입력값이 입력되면 입력값이 풀코드인지 아닌지를 시스템이 알 수 있는 것이다.

0> "14 … = bj …" 가 입력되면, 2번째 입력값 [4] 가 입력되는 순간 시스템이 입력값이 유효한 중국어 음절을 이루지 못하는 것(중국어 성모중 bj와 같이 로마알파벳 자음이 결합되는 경우가 없으므로)을 인식하고, 입력값을 심플코드로 간주하여 처리할 수 있다. 즉 도10-5에서 시스템이 심플코드 "14" 의 대응어구인 "Bejing" 혹은 "北京" 을 사용자에 제공할 수 있는 것이다. 만약 심플코드의 색인에 "14" 혹은 "14" 로 시작하는 심플코드가 없다면, 시스템은 입력값을 다시 풀코드로 간주하여 "bj" 를 사용자에게 제공하면 된다.

1> 이는 마치 모음요소를 활용한 한국어의 경우(도4-5)에서와 동일하다. 기 설명한 도4-5에서의 풀입력방법은 한국어제한 반복선택방법으로 볼 수 있고, 풀우선 병행입력방법 적용시, "12" 가 입력되면, 입력값이 유효한 한국어 음절을 이루지 못하므로, 시스템은 입력값을 심플코드로 간주하여 심플코드 색인을 참조하여 심플코드에 대응하는 어구를 사용자에게 제공하게 된다. 그러나, 심플코드 색인에 "12" 와 일치하는 심플코드가 더 이상 존재하지 않는 것이 확인되면, 시스템은 입력값을 다시 풀코드로 간주하여 "ㄱㄴ" 을 사용자에게 제공하게 된다.

22> 이해를 돕기 위하여 도5-4와 도10-5의 심플코드 색인에서 심플코드는 정렬되어 있는 상태로 표시하였으나, 시스템 내부에서 어떤 형태로 저장되어도 상관없으며, 다만 심플코드 색인 검색시 필요하면 시스템은 심플코드를 정렬하여 입력값이 심플코드 색인에 더 이상 존재하는지 여부를 검사하면 된다.

23> 또한 단축입력모드(검색대상이 되는 심플코드 색인에 도10-1을 기준으로 [1] ~ [9]로 이루어진 음절기준 이니셜코드만이 저장되어 있다고 가정)를 기본입력모드로 하는 병행입력방법(단축우선 병행입력방법)에서, 입력값 "1* ..." 에서 2번째 값 [*] 이 입력되는 순간 시스템은

심플코드 색인을 검색하지 않고도 입력값이 풀코드인 것으로 인지하여 처리할 수 있다. 이 역시 도5-4에서의 한국어의 경우와 동일하다.

1> 도5-4, 도10-5에서 병음/한자 색인과 심플코드 색인을 하나의 색인으로 설명하였다. 그러나 도5-1에서와 같이 병음/한자 색인(간단히 "병음색인"이라 부름. "병음 + 병음대응 한자"를 저장한 색인 - 입력된 병음으로부터 대응하는 한자 검색용)이 존재하고, 또 심플코드/병음 색인(간단히 "심플코드색인"이라 부름. "심플코드 + 심플코드대응 병음 + 병음대응 한자" 혹은 "심플코드 + 심플코드대응 병음"을 저정한 색인)이 별도로 둘 수도 있는 것은 자명하다.

5> 도5-4, 도10-5에서와 같이 심플코드 색인에 반드시 음절기준 이니셜코드만 저장되어 있을 필요는 없으며, 자음기준 이니셜코드와 같이 다른 유형의 심플코드가 함께 저장되어 있는 것도 가능하다. 마찬자기로 심플코드의 종류에 따라 음절기준이니셜코드색인, 자음기준이니셜코드색인, 전체연관심플코드색인 등과 같이 각각의 색인을 두어 구현될 수도 있다.

26> 도10-10에서와 같이 "北京"에 대하여 음절기준 이니셜코드 "14", 자음기준심플코드 "1473", 그리고 전체연관 심플코드 "1*#4#73"가 있다고 가정한다. 도10-10의 키패드에서 중국어제한 풀우선병행입력방법을 적용시, 음절기준 이니셜코드 "14" 그리고 자음기준 심플코드 "1473" 의 경우는 두번째 입력값 "7"이 눌러지는 순간 시스템이 입력값을 심플코드로 간주하고 색인을 검색하여 처리할 수 있다. 그런데, "beijing"의 전체연관 심플코드 "1*#4#73" 의 경우는 입력값 "1*#4 ..." 을 풀코드로 해석하면 "baiz ..."이 되고, "1*#4"까지는 병음의 생성규칙을 위반하지 않는다. 그러나 "baiz ..." (및 "baij ..." - 다음입력값에 따라 "baij ..."로 될 수 있으므로)은 도10-11의 병음색인에 존재하는 병음이 아니다. 따라서 시스템은 입력값 하

나하나의 입력에 따라, 입력값을 풀코드로 해석하여, 입력값이 "b... => ba... => bai... => baiz..." 로 각기 인식할 때마다 병음색인을 참조하여 일치하는 어구를 검색한다. 도10-10에서 보듯이 "bai~"까지는 일치하는 병음을 찾을 수 있으나, 입력값이 "baiz~" (및 잠재적 "baij~")로 인식되는 순간 시스템은 병음색인에 더 이상 일치하는 어구가 없는 것을 확인할 수 있게 된다(도10-10의 (1) 참조). 따라서 "baiz~" 까지의 입력값 "1*#4 ..."을 심플코드로 간주하고 심플코드 색인을 참조하여 일치하는 어구를 검색해 낼 수 있다 (도10-11의 (2)-(A), (2)-(B) 참조). 여기서 심플코드 색인에서도 입력값 "1*#4# ..." 과 일치하는 어구를 찾을 수 없으면, 시스템은 다시 입력값을 풀코드로 간주하여 "bai ..." 로 처리할 수 있다.

27> 뜻글자인 중국어의 특성상 모든 병음과 한자의 색인(병음/한자 색인)을 시스템에 저장하여야 하므로, 입력값이 유효한 병음음절을 이루더라도 병음/한자 색인에 존재하지 않는 것이 확인되는 순간 시스템은 입력값을 심플코드로 간주하여 처리하는 것을 의미한다. 이는 입력값이 풀코드를 이루지 못하는 것을 시스템이 인식하는 순간 입력값을 심플코드로 간주하여 처리하는 것과 같은 맥락으로 볼 수 있다.

28> 같은 단어에 대하여 다른 형태의 심플코드가 존재하는 경우 도10-10 에서와 같이 혹은 도10-11 에서와 같이 하나의 병음에 대하여 심플코드를 멀티플 카타로그의 어떤 형태로 가지고 있는 것이 모두 가능하다. 입력값을 심플코드로 간주하여 처리(즉, 입력값을 단축입력으로 간주하여 처리)함에 있어서, 도10-11에서의 (2)-(A) 혹은 (2)-(B) 의 처리가 선택적으로 가능하다. 도10-11의 (2)-(B)의 처리과정을 이용하면 색인에 심플코드를 가지지 않아도 된다. (2)-(B) 의 과정은 입력값 하나하나의 입력시마다 입력값이 이룰 수 있는 병음의 조합과 색인의 병음을 비교하여 일치하는 병음을 검색해 내는 것이다. 입력값이 이룰 수 있는 병음의 조합은 [1] 버튼에 2개 알파벳이 배정되어 있고 [*] 버튼에 3개의 알파벳이 배정되어 있으므로,

"1*"까지 눌러졌을때 각 버튼에 배정된 알파벳의 조합 6가지(=2x3) 가 가능함을 쉽게 알 수 있다. 이를 편의상 "심플코드 가능알파벳조합" 혹은 "가능알파벳조합"이라 부르기로 한다. 도10-11에서 [#] 버튼에 2개 알파벳이 배정되어 있으므로 "1*#"까지 눌러지면 12가지 조합이, "1*#4"까지 눌러지면 24가지 조합이 가능할 것이다. 그런데 시스템에서 구현시 실제 중국어 병음사전에서 "poi ..."로 시작하는 병음이 존재하지 않으므로, "poi..." 이하의 조합은 고려대상에서 제외할 수 있다. 즉, 입력값 "1*#4"에 대하여 "poi..."이하의 조합인 "poiz..." 및 "poij..." 도 제외될 수 있는 것이다. 마찬가지로 "poi..." 뿐 아니라 가능알파벳조합 중 병음색인에 존재하지 않는 조합을 제외하면서 입력값과 일치하는 병음을 검색할 수 있다.

29> 도10-11의 (2)-(A) 의 처리 과정을 이용하는 것은 색인에 심플코드를 가져야 하나, 불규칙하게 정의된 심플코드가 사용된 경우도 처리할 수 있다. (2)-(B)의 처리 과정을 이용하는 것은 심플코드를 색인에 저장하지 않아도 되나 일정한 규칙을 가진 심플코드(예를 들어, 전체연관 심플코드)가 입력된다는 전제가 필요하고 시스템에서는 그 약속된 심플코드 유형(예를 들어, 전체연관 심플코드)을 미리 알고 있는 것이 필요하다.

30> 입력하기를 원하는 모든 병음/한자의 색인을 가지고 있어야 하는 중국어의 특성상 도10-12, 도10-13에서와 같이 풀코드로 해석한 값이 색인에 존재하지 않을 때, 입력값을 심플코드(즉, 단축입력값)으로 간주하여 처리하는 것이 가능하다. 풀우선병행입력방법 적용시, 도10-12는 중국어 입력에 있어서 일반적으로 적용할 수 있는 절차를 나타내고 있다. 이는 음절기준 이니셜코드, 혹은 자음기준 심플코드를 사용하는 풀우선 병행입력방법에도 동일하게 적용될 수 있다. 예를 들어 음절기준 이니셜코드가 입력되면, 시스템은 이를 풀코드로 해석하여 병음색인을 검색

하게 되는데, 대부분의 경우 2~3번째 입력값이 입력되는 순간 입력값이 병음색인에 더 이상 존재하지 않는 것을 시스템이 인식할 수 있기 때문이다. 도10-13은 중국어의 언어적 특성을 이용하여 병음색인을 검색하기 이전에 입력값이 유효한 병음음절을 구성하는지를 체크하는 절차가 더해진 것을 보여준다.

31> 중국어뿐만 아니라 다른 언어의 입력시스템에서도, 입력하고자 하는 모든 단어의 색인을 시스템이 저장하고 있을 때, 마찬가지로 도10-12, 도10-13에서의 절차에서와 같이, 하나하나의 입력값이 입력될 때마다, 입력값을 풀코드로 해석하여 색인의 단어를 검색하여 일치하는지를 확인하고, 일치하는 단어가 더 이상 존재하지 않으면, 입력값을 단축코드로 간주하여 처리하는 것이 가능하다.

32> 14.1.2 영어 등

33> "st~" 혹은 "sp~" 로 시작하면서 최초 단어시작시부터 4번째 자음이 입력되는 것으로 인식되는 경우, 그리고 "st~" 혹은 "sp~"로 시작하지 않으면서 최초 단어시작부터 3번째 자음이 입력되는 것으로 인식되는 순간, 시스템은 입력값을 심플코드로 간주하여 처리할 수 있다.

34> 예를 들어 도1-1의 키패드에서 단순반복선택방법을 적용하고, “467... = gms ...” 입력시 세번째 입력값 [7]이 입력되는 순간 입력값이 영어의 단어생성규칙을 위배하므로, 시스템이 입력값을 심플코드로 간주하여 처리할 수 있는 것이다. 만약 심플코드 색인에 install 의 전체연관 심플코드 “4678255 = install” 가 저장되어 있다면, 심플코드 대응어구인 install 을 시스템이 사용자에게 제공할 수 있는 것이다. 검색범위의 심플코드 색인에 “467…” 과 일치하는 심플코드가 존재하지 않으면, 입력값을 다시 풀코드로 간주하여 처리할 수 있다.

35> 그리고 전술한 바와 같이 자모음분리키패드에서 영어제한반복선택방법을 풀입력방법으로 사용하면, 입력값이 영어의 입력규칙(예. 단어시작으로부터 "st~" 혹은 "sp~"로 시작하지 않으면서 3개의 자음이 올 수 없음)을 위배하는지를 쉽게 알 수 있으므로, 더욱 효율적으로 병행입력방법을 적용할 수 있다.

36> 14.2 대표알파벳제외 컨트롤처리방법을 풀입력방법으로 하는 언어제한 병행입력방법

37> 도1-1에서 각 버튼의 첫번째 알파벳을 대표알파벳으로 해당 버튼 1타로 입력하고, 나머지 알파벳을 컨트롤처리방법에 의하여 입력할 경우(즉, 대표알파벳제외 컨트롤처리방법 적용), 영어의 앞에서 설명한 단어생성규칙을 활용하여 단어생성규칙을 위배시 입력값을 심플코드로 간주하여 처리하는 언어제한 병행입력방법을 적용할 수 있다.

38> 예를 들어 도1-1에서 첫번째 알파벳을 대표알파벳으로 하고 나머지 알파벳을 컨트롤처리방법에 의하여 입력할 경우, "467... = gms ..." 입력시 세번째 입력값 [7]이 입력되는 순간 입력값이 영어의 단어생성규칙을 위배하므로, 시스템이 입력값을 심플코드로 간주하여 처리할 수 있는 것이다.

39> 마찬가지로 영어 이외의 언어에 있어서도 유사하게 적용할 수 있으며, 나머지 사항은 반복선택방법을 풀입력방법으로 하는 언어제한 병행입력방법에서와 유사하다.

40> 14.3 언어제한 병행입력방법에서의 3차모호성 극복

41> "scope" 의 전제연관 심플코드 "72673" 을 입력하는 경우에는, 시스템이 이를 1차적으로 풀코드로 인식하면 "72673 = pampd" 과 같이 되어 입력값이 영어의 단어생성규칙을 위반

하지 않으므로 시스템은 입력값을 1차적으로 풀코드로 인식하게 되고, 2차적으로 입력값을 심플코드로 해석하여 심플코드에 대응하는 scope 로도 인식하는 "3차모호성" 이 발생하게 된다. 여기서 풀입력방법으로 도1-1을 기준으로 단순반복선택방법이 적용된 경우이다.

<0012> 이 경우 시스템은 1차적으로 사용자에게 pampd 를 제공하게 되다. 그러나, 사용자가 scope 를 입력하고자 할 경우 특정 버튼(예를 들어, 아래화살표 [v] 버튼)을 눌러 scope를 선택하도록 할 수 있다. 또는 "72673" 이 입력되고 나서 단어입력이 종료되면(예를 들어 공백 등이 입력되면), 입력값을 1차적으로 풀코드로 해석한 "pampd" 와 심플코드로 해석한 "scope " 를 사용자에게 리스트로 제공하고 사용자로 하여금 선택하도록 할 수도 있다.

<0013> 위의 경우는 대표알파벳제외 컨트롤처리방법을 풀입력방법으로 사용하는 병행입력방법에서도 유사하게 적용할 수 있다. 또한 풀우선 병행입력방법이 아닌 단축우선 병행입력방법에서도 입력값 입력후 단어가 종료되는 시점에 입력값이 심플코드 색인에 존재하는 경우라도, 입력값을 풀코드로 간주하였을 때, 언어제한을 위반하지 않으면(3차모호성이 발생하면), 사용자로 하여금 선택(특정 버튼을 반복하여 눌러 혹은 리스트에서 선택)하도록 할 수 있다.

<0014> 영어 이외의 언어(특히, 대표알파벳제외 컨트롤처리방법을 적용하는 일본어의 경우 등)에 있어서도 유사하게 적용할 수 있다.

<0015> 다수의 후보 중 하나를 선택하기 위하여 이동버튼중 자주 사용되지 않는 임의의 버튼을 이용하여 시스템이 출력한 후보어구 중 타겟어구를 선택하도록 할 수 있다. 이는 3차 모호성이 발생하는 경우 뿐만 아니라 모호성이 발생하는 모든 경우에 적용할 수 있음은 이미 설명하였다.

15. 언어제한을 위배한 문자열을 이용하는 병행입력방법

입력값(풀코드 혹은 심플코드)는 클라이언트측에서 해석될 수도 있고, 서버측에서 해석될 수도 있다고 하였다. 도11-1, 도11-2는 각각 클라이언트측에서 해석되는 경우와 서버측에서 해석되는 경우를 나타낸 선출원의 도면이다. 즉 단순히 "시스템"이라 하면 클라이언트측 시스템 혹은 서버측 시스템을 모두 아우르는 개념인 것이다. 입력값이 심플코드인 경우 이를 클라이언트측에서 해석하여 심플코드 대응어구를 문자열(text)로 서버측에 전송하는 경우, 서버측에서는 이를 활용하여 각종의 서비스를 제공하면 된다. 마찬가지로 입력값을 숫자로 서버측에 전송하고, 서버측에서 해석하는 경우도, 서버측에서 심플코드를 해석하여 각종의 서비스에 활용하면 된다. 클라이언트측에서 DTMF 톤을 송출하고 이 입력값을 서버측에서 전달받아 활용하는 것도, 당연히 숫자값을 클라이언트측에서 서버측으로 전송하는 사례중의 하나이다.

언어제한 병행입력방법에서 입력값이 언어제한을 위배하여(즉, 입력값이 특정언어의 유효한 음절을 이루지 못하여), 시스템이 입력값을 심플코드로 간주하여 처리하되, 심플코드의 색인에 입력값과 일치하는 심플코드가 존재하지 않을 수도 있다. 이 경우 다시 입력값을 풀코드로 간주하게 된다고 하였다. 그런데 예를 들어 사용자가 도4-5를 기준으로 "삼성전자"를 염두에 두고 "7799"를 입력하였는데, 클라이언트측 시스템내의 심플코드 색인에 "7799" 와 "7799"에 대응되는 "삼성전자"를 저장하고 있지 않으면, 시스템은 단순히 "ㅅㅅㅈㅈ"을 사용자가 입력한 것으로 간주하게 된다. 만약 증권정보시스템의 경우, 클라이언트측에서 입력값 "7799"를 서버측으로 전송하거나, 혹은 입력값 "7799"에 대응되는 삼성전자를 서버측으로 전송하면, 증권정보서버에서 "7799"를 해석할 수 있으므로 별 문제가 없을 것이다(증권정보 서버에서는 심플코드 "7799"와 심플코드 대응어구인 "삼성전자"를 가지고 있는 것으로 가정). 그러나 클라

이언트측에서 문자입력모드에서 사용자가 "7799"를 입력후 클라이언트측에서 입력값을 심플코드로 간주하여 처리하고, 클라이언트측 심플코드 색인에 "7799"가 존재하지 않아 이를 다시 풀코드로 간주하여 문자열 "ㅅㅅㅈㅈ"을 서버측에 전송하는 경우 증권정보 서버에서는 이를 활용하기 어렵게 된다.

<19> 따라서 증권정보 서버에서는 전송된 문자열 "ㅅㅅㅈㅈ"가 의미를 가지는 값이 아닌 경우, "ㅅㅅㅈㅈ"를 입력하기 위하여 사용된 "7799"를 추출하고, 심플코드 "7799"의 대응어구인 "삼성전자"를 추적하여 이를 시스템에서 활용할 수 있다. 혹은 한국어에서 심플코드로 음절기준 이니셜코드를 사용할 경우만 적용할 수 있는 다른 방법으로, 시스템은 심플코드 대응어구 중 "ㅅㅅㅈㅈ"과 음절의 첫자음이 일치하는 "삼성전자"를 추적하여 이를 활용하여 서비스를 제공할 수 있다. "쌈썽쩐차" 의 경우도 음절기준 이니셜코드는 "7799" 가 된다. 이 경우도 "7799" 는 "ㅅㅅㅈㅈ" 으로 해석되는데, "쌈썽쩐차" 를 검색해 내기 위해서는 첫자음이 동일한 경우를 포함하여, "ㅅ" 의 경우는 음절의 첫자음이 "ㅆ" 인 경우까지, "ㅈ" 에 대해서는 "ㅉ" , "ㅊ" 인 경우까지 포함하여 검색하면 된다. 즉, "ㅅ" 과 함께 배정된(격음, 경음이 기본자음이 배정된 버튼에 암시적으로 배정된 것으로 간주) 알파벳이 음절의 첫자음과 일치하는 경우를 검색하면 된다.

<50> 이는 키패드가 아닌 키보드를 사용하는 PC 환경에서도 동일하게 적용될 수 있다. 즉 기존의 PC 등을 클라이언트로 사용하는 증권정보 시스템에서 사용자는 "삼성전자"의 주가를 조회하기 위하여 "삼성전자"를 입력여야 했지만, "ㅅㅅㅈㅈ"만을 입력하고 서버측으로 전송하더라도, 증권정보 서버 시스템에서 "ㅅㅅㅈㅈ"이 낱글자의 나열로써 상장회사명에 존재하지 않는 것을 인지하고, 앞서 설명한 2가지 방법("7799"를 추출 후 "삼성전자"추적 혹은 "ㅅㅅㅈㅈ"와 음절의 첫자음이 일치하는 "삼성전자" 추적)에 의하여 서비스를 제공할 수 있는 것이다.

1> 또 다른 방법은 서버측 색인에 "7799(심플코드), 삼성전자(심플코드 대응어구)" 뿐만 아니라 "ㅅㅅㅈㅈ(편의상 '심플코드 대응문자' 라고 함)"를 등록하여 두고, 클라이언트측으로부터 전송된 값 "ㅅㅅㅈㅈ"이 유효한 한국어 음절을 이루지 못하는 것을 확인하고 저장된 심플코드 대응문자와 비교하여 사용자가 의도하는 것이 "삼성전자" 라는 것을 알아낼 수 있다.

2> 이상의 내용을 그림으로 정리하면 도11-3과 같으며, (A), (B), (C)가 선택적으로 가능하다.

3> 더 나아가 기존의 PC 등의 클라이언트에서 키보드 등과 같은 장치를 통하여 문자를 입력하는 등의 경우(예를 들어, PC의 워드프로세서 등을 이용하여 문자를 입력하는 경우)에 있어서도, 음절기준 이니셜코드를 이용하여 음절의 첫자음만을 입력하고, 시스템(클라이언트 혹은 서버)에서 이를 타겟어구로 변환함으로써, 고속으로 문자입력을 할 수 있도록 하거나 혹은 각종 정보시스템에서 활용하는 것이 가능하다. 예를 들어 "삼성전자"를 입력하고자 할 경우 사용자가 "ㅅㅅㅈㅈ"을 입력하면, 시스템은 색인을 검색하여 음절의 첫자음이 "ㅅㅅㅈㅈ"인 "삼성전자"를 사용자에게 제공할 수 있다. 앞에서 설명한 다른 방법(약속된 특정 키패드를 기준으로한 입력값의 심플코드 추출 및 색인검색, 심플코드 대응문자를 색인에 저장)도 적용이 가능하다. 도11-5를 참고할 수 있다. (A), (B), (C)가 선택적으로 가능하다.

4> 지금까지는 상용어구를 등록하여 두고 이를 입력함에 있어서, 단순히 어구와의 연관성과 관련없이 단순히 특수기능버튼과 숫자버튼 등을 조합하여 입력하도록 지정하는 정도였다. 예를 들어 "삼성전자"를 상용어구로 등록하고, "alt + 1"로 입력하도록 하는 것이다.

5> 마찬가지로 도10-2을 기준으로 "四通集團(satongjipdan : 출원인이 중국어 발음을 알지 못하는 관계로 편의상 한국어 발음을 한어병음인 것으로 간주하여 사용하나, 중국어 발음에도 유사하게 적용하면 됨)" 의 음절기준 이니셜코드는 s, t, j, d 에 대응되는 "6242" 가 된다. 사용자가 "6242" 를 입력하였을 때, 클라이언트측 시스템의 심플코드 색인에 6242와 四通集團(satongjipdan)이 등록되어 있지 않으면, 클라이언트측 시스템에서는 사용자에게 단지 "sdzd" 만을 제공하게 된다. 사용자가 "sdzd" 를 서버측으로 전송하면, 서버측에서는 " sdzd" 가 유효한 중국어 음절을 이루지 못하므로, "sdzd" 로부터 사용자가 입력한 입력값 " 6242" 를 추출하고, 심플코드(여기서는 음절기준 이니셜코드) "6242" 에 대응되는 "四通集團" 을 검색해낼 수 있는 것이다(서버측 시스템에서는 심플코드와 대응어구를 저장하고 있는 것으로 가정).

56> 한국어의 사례에서 "ㅅㅅㅈㅈ" 및 각 자음과 함께 배정된 알파벳과 각 음절의 첫자음이 일치하는 어구를 검색해내는 것과 처럼, 도10-2을 기준으로 "sdzd" 및 기타의 변형이 가능한 자음조합(예. "xdjd" , "stjt" , "sdjt" , … 16가지 경우 존재)과 각 음절의 첫자음이 일치하는 단어를 사용자에게 제공할 수 있다.

57> 역시 서버측에 심플코드 대응문자 "sdzd" 를 함께 저장하고, 클라이언트측으로부터 전송된 값 "sdzd" 이 유효한 중국어음절을 이루지 못하므로, 전송값과 색인의 심플코드 대응문자를 비교하여 사용자가 의도하는 것이 "四通集團" 인 것을 시스템이 알아낼 수 있다.

58> 이상의 내용을 그림으로 정리하면 도11-4와 같으며, (A), (B), (C)가 선택적으로 가능하다.

도11-3, 도11-4에서의 처리과정은 서버측에서 행해질 수도 있고 클라이언트측에서 행해질 수 있다.

키패드를 구비한 장치뿐만 아니라 PC등의 정보통신기기에서도 사용자가 "stjd" 입력시 한어병음의 음절의 첫자음이 일치하는 단어("四通集團")을 검색하여 사용자에게 제공할 수 있다. 혹은 사용자가 "stjd" 입력시 시스템(예. PC)에서 단축코드(심플코드 대응문자)를 각 음절의 첫자음 "stjd" 로 등록하여 두고, 두번째 자음이 입력되는 순간, 중국어 언어제한을 위배(sh, ch, zh 등이 아닌 2개의 로마알파벳 자음 등장)하므로, 입력값을 단축입력값으로 간주(사용자가 단축입력을 하는 것으로 간주)하여 처리할 수 있다. 입력값 "stjd" 에서 심플코드를 추출하는 것은 미리 약속된 키패드(예. 10-2)에 의하여 심플코드(예. 6242)를 추출할 수 있다. 도11-6을 참고할 수 있으며, (A), (B), (C)가 선택적으로 가능하다.

도11-6의 (B)는, 입력값이 생성하는 그리고 유효한 병음 음절을 생성하지 않는 단어 (예에서는 "음절기준이니셜값"으로 "stjd")와 색인에 저장된 어구의 각 음절의 첫 알파벳 (" sa-tong-jip-dan => stjd") 과 비교하여 일치하는 어구를 검색해 내는 것이다. 이는 단축입력값으로 음절기준이니셜값을 사용하는 것으로 볼 수 있다. 도10-*의 키패드에서는 "stjd"에 대응하는 음절기준이니셜값은 "622442"이 된다. 도11-6의 (B)는 도면에서 쉽게 알 수 있듯이, 심플코드 색인을 구비하지 않고, 단지 한자의 입력을 위한 병음의 색인을 구비하는 것만으로 가능하다. 마찬가지로 입력값이 "자음기준입력값"이 사용되는 경우에도 시스템에서는 입력값이 생성하는 단어(문자열)와 색인에 저장된 어구의 자음이 일치하는 어구를 검색해 낼 수 있다.

정리하면 규칙적으로 인식될 수 있는, 단축입력하는 입력값의 종류는 심플코드의 종류와 유사하게 "음절기준이니셜값"(예를 들어, "beijing 의 음절기준이니셜값은 "144 = bj" 가 됨),

"자음기준입력값" ("beijing" 의 자음기준입력값은 "144773 = bjng"가 됨), "첫모음+음절기준이니셜값", "단어기준이니셜값" 등등이 있을 수 있다. 시스템은 입력값이 언어제한을 위배하는 순간 입력값을 단축입력값으로 간주하여 처리하는 것이다.

<33> 지금까지 제시한 "단축입력값"의 종류를 정리하면 도11-11의 표와 같다. 도11-11에서 단축입력값 (A)는 (B) ~ (L)까지를 포괄하며, 이는 지금까지 제시하였던, 시스템이 일정한 규칙에 따라 인식할 수 있는 단축입력값의 종류를 보여준다. 임의의 일부 알파벳에 연관하여 정해진 일부연관심플코드 그리고 임의의 일부 알파벳만으로 이루어진 단축입력값도 포함된다. "전체입력값" (Z)는 풀입력방법의 풀코드가 됨을 알 수 있다.

<34> 기 제시한 대로 어떤 조건(예를 들어, 입력값이 풀코드의 생성규칙을 위반하는, 또는 특정언어의 유효한 음절을 이루지 못하는 경우 등등)에 해당하면 입력값을 단축입력값으로 간주하여 처리하는 것 (즉, 입력값이 약속된 풀입력방법의 풀코드가 아닌 것으로 간주하여 처리)이, 병행입력방법의 핵심내용이므로 본 발명에서 제시하지 않은 단축입력값이 사용되는 경우도 본 발명의 병행입력방법의 범주에 포함됨은 자명하다. 도11-11의 표에서 (H) ~ (L)의 단축입력값은, 도11-5, 도11-6에서와 같이, 키패드를 구비한 장치 뿐만아니라 PC와 같은 장치에서도 이용될 수 있다. 도11-6의 (B)는 단축입력값으로 "음절기준이니셜값"이 사용된 사례를 보여준다. 단축입력값을 통칭하는 용어로 심플코드 혹은 단축코드 라는 용어가 사용되었는데, (H) ~ (L) 의 단축입력값 역시 포괄적인 의미의 심플코드로 볼 수 있다. 즉 도11-11에서의 (I) "음절기준이니셜값"은 "음절기준이니셜값 심플코드"라고 할 수 있을 것이며, (H) ~ (L) 의 단축입력값 역시 마찬가지로 명명할 수 있을 것이다.

<65> 도10-*에서 음절기준이니셜값으로 "14 = bz"가 입력되면 시스템은 "bz"와 각 음절의 자음이 일치하는 어구를 색인에서 검색해 낼 것이다. 여기서 다시 "4"가 입력되면 "144 = bj" 이므로 "bj"와 각 음절의 첫자음이 일치하는 어구(예를 들어, "beijing")를 색인에서 검색해 낼 것이다.

<66> 본 발명에서는 한걸음 더 나아가 앞에서 예시한 단축입력값 중 임의의 단축입력값이 입력되었을 때, 이를 여러 종류의 단축입력값으로 해석할 수 있음을 보인다. 예를 들어, 어떤 단축입력값에 대하여 1차적으로 소정의 단축입력값(예를 들어, 음절기준이니셜값)으로 간주하여 처리하고, 2차적으로 다른 단축입력값(예를 들어, 자음기준입력값)으로 간주하여 처리하고, 3차적으로 또 다른 단축입력값으로 간주하여 처리하고, 이하 하나의 단축입력값에 대하여 여러 종류의 단축입력값으로 해석할 수 있음을 제시한다. 도10-*을 기준으로 단축입력값 "144 = bj" 가 입력되면 시스템은 1차적으로 "bj"와 각 음절의 첫 알파벳이 일치하는 어구(예를 들어, "beijing")를 색인에서 검색하고, 1차 검색결과가 존재하는지 여부와 상관없이, 2차적으로 색인의 어구중 자음이 일치하는 어구를 검색할 수 있는 것이다.

<67> 이는 사용자가 도11-11에서 제시한 어떤 단축입력값을 입력하더라도 시스템이 해석해 낼 수 있음을 보여주는 것이다. 이를 적용하면 하나의 단축입력값에 대하여 다수의 검색결과가 있을 수 있게 되는데, 출력되는 우선순위는 적용하는 단축입력값의 유형에 따라 우선순위를 적용할 수 있다. 예를 들어 음절기준이니셜값으로 해석하여 검색된 어구(들)을 우선으로, 다음으로 자음기준입력값으로 해석하여 검색된 어구(들)을 그 다음으로 출력하는 것이다. 만약 어떤 사용자가 단축입력값으로 음절기준이니셜값을 주로 사용한다면, 음절기준 이니셜값에 의하여 해석된 어구(들)을 1차적으로 출력되도록 설정하는 것이 바람직할 것이다. 따라서 단축입력값의 유형에 따른 우선순위를 사용자가 지정할 수 있도록 하는 것이 바람직하다. 또한 어떤

사용자가 자음기준이니셜값 만을 사용하기를 원한다면 단축입력값을 자음기준 이니셜값으로만 해석하도록 설정할 수 있는 것이 바람직할 것이다. 같은 유형의 단축입력값으로 해석하여 얻어진 어구들의 출력 우선순위는 이미 언급한 바와 같이 사용빈도와 같은 요소에 의하여 결정될 수 있다. 또한 출력되는 우선순위는 단축입력값 해석에 적용된 단축입력 유형에 관계없이 검색된 어구들 각각의 사용빈도에 따라 결정될 수도 있다.

<68> "jinzhi (禁止) 과 zhuijian(逐ㅇ) 의 경우는 음절기준 이니셜값 각각 "jz", "zj"로 모두 "444" 이다. 이 경우는 풀우선 병행입력방법 적용시, 3번째 입력값 "4"가 입력되는 순간 입력값이 병음음절을 이루지 못하는 것을 시스템이 인지할 수 있게 된다. 도10-* 을 기준으로 음절기준이니셜값이 같은 버튼에 배정된 각기 다른 알파벳인 경우는 가능한 모든 경우를 검색하면 된다. 즉 단축입력값 "444"에 대하여 각 음절이 "jz"와 일치하는 어구 미치 "zj"와 일치하는 어구를 모두 검색하면 된다.

<69> 마찬가지로 단축우선 병행입력방법 적용시에도, 입력값을 도11-11의 (B) ~ (L)의 단축입력값으로 간주하여 처리할 수 있다. 또한 입력값을 1차적으로 음절기준이니셜값으로 간주하여 처리하고 2차적으로 음절기준이니셜코드로 간주하여 처리하고, 3차적으로 자음기준이니셜값으로 간주하여 처리하고, 이하 마찬가지로 다른 단축입력값으로 처리하는 것이 가능하다. 입력값을 단축입력값으로 처리하여 더 이상 일치하는 단어가 색인에 존재하지 않는 것이 확인되는 순간 시스템은 입력값을 풀코드로 간주하여 처리하면 된다. 사용자가 주로 사용하는 단축입력값의 종류를 미리 지정할 수 있도록 하는 것이 바람직할 것이다.

<70> "jinzhi (禁止) 과 zhuijian(逐ㅇ) 의 경우는 음절기준 이니셜값을 사용할 때의 문제를 해결하기 위하여 권설음의 음절기준이니셜값 심플코드 및 음절기준 이니셜코드를 다음과 같이 할 수 있다. 중국어에서 권설음 "ch", "sh", "zh" 는 사실상 단일 성모이고, 주음부호

로는 하나의 기호로 표기되나, 영어알파벳으로 표기시 2개의 알파벳 조합으로 된 것이다. 따라서 도10-6에 따른 zhuijian 의 음절기준이니셜값 심플코드 역시 "h" 를 포함하여 "zh+j" 에 대응되는 "4944" 로 할 수 있다. 음절기준이니셜코드는 "zh+j" 에 대응되는 "494" 가 될 것이다. 도10-*의 키패드에서 권설음 "ch", "sh", zh" 를 이루는 첫 알파벳 "c", "s", "z" 가 두번째 알파벳 "h" 와 다른 버튼에 배정되어 있으므로, "jinzhi (禁止)" 과 "zhuijian(逐ㅇ)" 의 음절기준이니셜값 심플코드를 모두 "444" 로 하였을 때의 모호성이 없게 된다.

71> 음절기준이니셜값 심플코드를 단축입력값으로 이용하는 풀우선 병행입력방법에서, 입력값 "4~" 는 "z~" 로 인식되고, 입력값 "49~" 가 입력되면 "zh~" 로 인식하고 처리하게 될 것이다. 그러나 "zh~" 로 시작하는 단어는 다수 이므로 시스템이 입력값 "49~" 를 "zh~" 로 처리하여 대응어구를 사용자에게 출력하더라도 의미가 적을 것이다. 따라서 입력값 "49" 에 대하여 시스템은 음절기준이니셜값 심플코드로 해석하여 음절의 구성이 "z__h__" 인 어구를 출력하여 제공할 수 있다 (밑줄 부분은 모음이 위치함을 의미). 만약 "49" 를 "zh" 로만 해석하면, 음절기준 이니셜값이 "z__h__" 인 단어를 단축입력하기 어렵게 된다. 중국어의 경우 대분분의 단어가 하나의 음절(글자) 혹은 두개의 음절(글자)로 이루어져 있으므로, 2음절 단어의 단축입력을 위하여, 권설음 입력에 대하여 단축입력으로 처리하는 것이 필요하다. 도10-17을 참고한다. 도10-17에서 액정상에 [禁止] 다음에 흐린색으로 [. . .] 으로 표시된 것은 "z__h__" 인 단어가 다수인 경우 후순위의 다른 단어들이 출력됨을 의미한다. 후순위로 "zh__" 인 단어들이 출력될 수 있음은 물론이며, 사용자의 취향에 따라 "zh__" 인 단어가 먼저 출력되도록 설정할 수도 있을 것이나, 일반적으로 "z__h__" 형태의 단어가 먼저 출력되도록 하는 것이 편리할 것이다.

2> “49” 다음에 모음 입력이 인식되면(예를 들어, 모음버튼이 눌러지면), 시스템은 입력값을 “zh__” 로 처리(즉, 입력값을 풀코드로 처리)하게 된다. “49” 다음에 자음 입력이 인식되면(예를 들어, 자음버튼이 눌러지면), 시스템은 입력값을 “zh__X__” 로 처리하게 된다(대문자 ‘X’ 는 임의의 성모를 표현하는 영어 자음). 즉 정리하면 풀우선 병행입력방법에서도 권설음의 “ch”, “sh”, “zh” 등의 입력에 대하여 일차적으로 심플코드(음절기준이니셜값 심플코드 혹은 음절기준 이니셜코드)로 해석하고, 다음 입력값이 모음인지 자음인지에 따라 풀코드 혹은 심플코드로 해석하는 것이다.

73> 위의 내용은 자음기준이니셜값 심플코드를 단축입력값으로 하고, 단축우선병행입력방법을 적용한 것과 유사한 결과이다. 단, 단축입력값으로 음절기준 이니셜값 심플코드를 적용하고, 풀우선 병행입력방법을 적용하되, 단지 단어 시작상태 (확인버튼이 눌러지거나 혹은 입력시스템에서 약속된 형태의 단어종료 후)에서 권설음이 입력되었을 때는 입력값을 단축입력값으로 간주하여 처리(예에서, 단어의 초성이 “z__h__” (‘ch’ , ‘sh’ 등의 다른 권설음의 경우는 그에 따름) 와 일치하는 단어를 검색)하는 것이다.

74> 이상에서 도10-*을 기준으로 설명하였으나, 이는 다른 키패드에서도 위의 개념을 적용할 수 있음은 자명하다.

75> 더 나아가 PC 의 키보드 그리고 키패드 뿐만 아니라 이와 유사한 변형된 모든 형태의 자판(스크린 상에서 구현되는 자판의 형태는 매우 다양할 수 있다)에서도 동일하게 적용할 수 있다.

'6> 16. 심플코드 가능알파벳 조합을 이용하는 병행입력방법

'7> 앞에서 설명한 방법중 심플코드로 음절기준 이니셜코드를 사용하는 경우, 각 음절의 첫 자음과 비교하는 것은, 클라이언트측에서 심플코드의 색인(예. 6242 / satongjipdan / 四通集團)을 가지지 않고, 단지 특정 어구의 색인만(예. satongjipdan / 四通集團)을 가진 경우에도 유용하게 적용할 수 있다. 예를 들어 "62…" 에서 두번째 [2] 가 입력되는 순간, 시스템은 입력값이 중국어 언어제한을 위배하는 것을 인식하고 입력값, "62… = sd…, st…, xd…, xt …(편의상 '가능알파벳조합' 이라 부름)" 과 색인의 단어 중에서 각 음절의 첫자음이 일치하는 어구를 검색하여 사용자에게 제공할 수 있는 것이다. 버튼 누름의 횟수가 많아지면, 가능알파벳조합의 수가 더 많아지게 되며 이 가능알파벳조합을 이용하여 색인의 어구와 비교/검색하면 된다. 도11-7을 참고할 수 있다. 도11-7의 색인에서 "四通集團" 행(record)에서 다중(multiple)적으로 "s t j p" 및 "s d .." , "x d …" , "x t …" 의 목록(2**4 = 16가지가 가능함) 을 가지고 있으면서, 입력값 "sd …" 에 대하여 각 레코드의 이 다중값과 비교하여 사용자에게 四通集團 을 제공할 수도 있을 것이다. 이 역시 같은 개념이나 구현이 어렵고, 장점이 없으므로 권장하지 않는다. 도11-9를 참고한다. 어느 방법이나 입력값으로 표현될 수 있는 가능알파벳조합과 색인의 어구(도11-9에서는 병음)을 비교하는 것은 같다.

578> 마찬가지로 클라이언트측에서 심플코드 색인(예. 7799, 삼성전자)을 가지지 안고, 단지 특정 어구의 색인(예. 삼성전자)만을 가지고 있는 경우, 사용자가 "삼성전자" 를 염두에 두고 , "77…" 를 입력시, 시스템은 2번째 [7] 이 입력되는 순간, 입력값이 한국어 언어제한을 위배하는 것을 인식하고, 입력값 "77… = ㅅㅅ…, ㅅㅆ…, ㅆㅅ…, ㅆㅆ… (편의상 '가능알파벳조합' 이라 부름)" 과 색인의 단어 중에서 각 음절의 첫자음이 일치하는 어구를 검색하여 사용자에게 제공할 수 있는 것이다. 도11-8을 참고할 수 있다.

29> 만약 입력값과 일치하는 어구가 색인에서 더 이상 존재하지 않는 것이 확인되면, 입력값을 풀코드로 간주하여 사용자에게 제공하면 된다. 이는 사용자가 음절기준 이니셜코드의 값만을 입력한다는 전제하에, 색인에 심플코드를 두지 않고, 단지 단어의 색인만을 둔 상태에서 병행입력방법과 유사한 효과를 낼 수 있는 것이다. 단, 이 방법 역시, 풀우선 병행입력방법에서 입력값이 풀코드 생성규칙을 위배하거나, 풀우선 언어제한 병행입력방법에서 입력값이 언어제한을 위배하는 순간을 시스템이 감지하는 절차까지는 동일하며, 입력값에 대응되는 어구를 검색해내는 절차만 다른 것이다. 즉, 풀코드 생성규칙을 위반하거나 언어제한을 위배하는 순간, 시스템이 입력값을 풀코드가 아닌 것으로 간주하되,심플코드를 이용하지 않고 입력값이 음절기준 이니셜코드인 점을 이용하여 입력값에 대응되는 어구를 해석(혹은 검색)하여 사용자에게 제공하는 것이다.

30> 병행입력방법의 핵심적인 내용은 풀코드 생성규칙을 위반하는 순간, 혹은 언어제한을 위반하는 순간을 시스템이 자동으로 감지하여, 입력값을 심플코드로 간주(즉, 사용자가 풀입력이 아닌 단축입력을 하는 것으로 간주)하는 것이었고, 이는 결국 기 제시한 병행입력방법에서 입력값에 대응되는 어구를 검색해내는 부분만 특별한 경우(음절기준 이니셜코드 이용)에 한정하여 변형한 것이다.

31> 음절기준 이니셜코드의 경우와 마찬가지로, 전체연관 심플코드, 자음연관 심플코드, 첫모음+자음연관 심플코드 등과 같이 규칙성을 가지고 있어서 자동으로 심플코드를 추출할 수 있는 경우에는 입력값으로 부터의 "가능알파벳조합"과 비교하여 색인의 어구를 비교하여 색인으로부터 타겟어구를 검색하는 것이 가능하다. 도10-11에서 시스템이 입력값 하나하나의 입력시마다 입력값을 풀코드로 해석하여 풀코드로 해석된 병음이 병음색인에 존재하지 않는 것을 확인하는 순간, 시스템은 입력값을 단축입력값으로 간주하여 입력값에 대한 "가능알파벳조합" 과

병음을 비교함으로써 타겟어구를 검색할 수 있는 것이다. 이를 도면으로 표시하면 도11-10와 같이 되며, 이는 도10-11의 일부분((2)-(B)부분)를 표현한 것으로 전체연관심플코드가 사용된 경우의 사례이다.

<2> 17. 풀코드를 저장하고 있는 입력시스템에서의 풀입력방법 및 병행입력방법

<3> 특정 키패드에서의 특정 언어제한 풀입력방법에 의하여, 입력값에 대한 올바른 단어를 식별해내는 것이 조금은 복잡한 경우가 있음을 전술한 내용에서 알 수 있다. (예. 표준영어키패드에서 반복선택방법에 의한 영어 입력 및 중국어 병음 입력) 앞서 중국어에서 설명한 바와 같이, 중국어 뿐만 아니라 다른 언어에서도 입력하고자 하는 모든 단어의 색인을 시스템이 가지고 있는 경우, 입력값을 1차적으로 풀코드로 해석하고, 색인에 존재하지 않을 경우 이를 2차적으로 약속된 심플코드로 해석하는 것이 가능하다.

<84> 이하에서 편의상 중국어의 경우를 예로 설명한다. 도10-14에서와 같이 중국어 병음(타언어의 경우 가능한 단어)을 모두 저장하고 있는 시스템에서 풀코드를 모두 저장하고 있으면, 시스템은 단순히 입력값과 일치하는 어구를 검색하여 타겟알파벳을 인식해 낼 수 있다. 만약 풀우선 병행입력방법을 적용한다면, 일치하는 풀코드 값이 색인에 더 이상 존재하지 않는 것을 확인하는 순간 시스템은 입력값을 약속된 유형의 심플코드로 간주하여 처리하게 된다. 도10-14에서 "北京"을 입력하는 과정은 입력값을 풀코드로 해석하여 처리하는 과정으로 볼 수 있다. 단지 풀코드를 색인에 저장하고 있으므로 색인의 검색을 통하여 풀코드를 해석하는 것이다.

<85> 언어제한 입력방법에 의하여 모호성없이 단어가 식별될 수 있는 풀입력방법을 적용하는 경우, 색인에 풀코드를 저장하고 있는 것이 유용하지 않다. 그러나 반복선택방법과 같이 모호성있는 풀입력방법을 적용하고, 입력하고자 하는 어구의 모든 색인을 저장하고 있는 경우 유용

할 수 있다. 앞서 "불완전 자모음분리키패드에서의 언어제한입력방법" 에서 설명한 바와 같이 모호성이 발생할 수 있는 풀입력방법을 적용시 입력하고자 하는 모든 단어가 색인에 저장되어 있다면(중국어 뿐만 아니라 다른 언어의 경우에도 적용될 수 있음), 이 풀코드 색인을 이용하여 입력값을 처리할 수 있다.

<6> 도1-1의 표준영어키패드에서 중국어제한 반복선택방법을 적용하여 중국어 병음 입력시, "~ngh+모음~" 의 경우는 "~ni+모음" 으로 인식될 수 있는 경우가 있다고 하였고 이 경우는 두가지 모두에 대하여 일치하는 어구를 병음색인에서 검색하여 처리할 수 있다고 하였다. 병음색인에 풀코드(도1-1을 기준으로한 반복선택방법에서의 풀코드)를 모두 저장하고 있는 경우, 중국어제한 반복선택방법을 적용하는 것과 마찬가지로 입력값이 생성할 수 있는 모든 경우(예. "~ngh+모음" 및 "~ni+모음")에 대하여 색인을 검색하여 일치하는 어구를 시스템이 인식해 낼 수 있는 것이다.

<37> 예를 들어 도1-1, 도5-1에서 반복선택방법 적용시, "222" 는 "aaa" , "ab" , "ba" , "c" 로 인식될 수 있다. 따라서 도1-1, 도5-1에서 "beijing = 22334445444664" 이고, 여기서 연속된 숫자그룹은 다수의 알파벳 조합으로 인식될 수 있다. 연속된 숫자 그룹인 "22~" 는 "aa" 혹은 "b" 로 해석될 수 있다. "~33~" 은 "dd" 혹은 "e" 로 해석될 수 있다. 따라서 "2233~" 은 "aadd~" , "aae~" , "bdd~" 또는 "be~" 의 4가지로 해석될 수 있다. 입력값이 증가하면 가능 조합의 가짓수는 기하급수적으로 증가하게 된다. 이렇게 입력값에 대하여 해석가능한 알파벳 조합을 편의상 역시 "가능알파벳조합" 혹은 "후보 알파벳조합" 이라 부른다.

<88> 그러나, "22~" 가 눌러졌을 때 "aa~" 혹은 "b~" 로 해석될 수 있는데, 색인을 참조하여 "aa~" 로 시작하는 단어가 더 이상 색인에 존재하지 않는 것을 시스템이 확인하면, 시스

템은 "aa~" 로 시작하는 조합은 제외하고, 입력값 "22~" 를 "b~" 로 처리할 수 있게 된다. 이는 중국어 병음이 "aa~" 로 시작하지 않는 중국어 제한을 이용하여 입력값 "22~" 를 "b~" 로 해석할 수 있는 것과 유사하게 볼 수 있다. 따라서 "2233~" 이 눌러졌을 때, "bdd~" , "be~" 의 2가지 조합에 대해서만 색인을 검색하게 되고, 여기서 다시 "bdd~" 로 시작하는 어구가 색인에 더 이상 존재하지 않는 것을 시스템이 확인하면, "bdd~" 로 시작하는 조합은 더 이상 고려하지 않고, "be~" 만을 대상으로 다음 입력값에 따라 처리하면 된다. 역시 중국어 병음에서 "bdd~" 가 불가능한 중국어제한을 이용하여 "입력값 "2233~" 을 "be~" 로 처리하는 것과 유사하게 볼 수 있다. 입력값 "22334~" 에 대하여 "beg~" 로 해석될 수 있다. 또 입력값 "223344~" 에 대하여 "begg~" 혹은 "beh~" 로 해석할 수 있는데, 병음 색인을 검색하여 "begg~" 로 시작하는 단어가 더 이상 존재하지 않는 것을 확인하는 순간 시스템은 입력값 "223344~" 를 "beh~" 로 처리하게 된다. 이는 중국어제한 입력방법(예. 중국어 병음 생성규칙에 의거하여 "begg~" 가 유효한 음절이 아니고 "beh~" 만이 유효한 병음 음절을 이루는 중국어제한을 이용한 입력방법)을 이용하는 것과 유사하게 볼 수 있다. "4" 가 한번 더 눌러져서 입력값이 "2233444~" 가 되면, "behg~" 혹은 "bei~" 로 해석될 수 있는데, 역시 색인을 검색하여 "behg~" 로 시작하는 단어가 더 이상 존재하지 않는 것을 시스템이 확인하는 순간 입력값을 "bei~" 로 처리할 수 있으며, 이는 중국어제한 입력방법에서 "behg~" 가 불가능한 병음 조합이므로 입력값 "2233444~" 를 "bei~" 로 해석하여 처리하는 것과 유사하게 볼 수 있다.

589> 입력하고자 하는 모든 단어와 풀코드를 저장하고 있는 시스템에서, 단순히 "beijing = 22334445444664" 에서 후보 알파벳조합의 가짓수는 2*2*4*1*4*2*1 = 128 가지이다. 물론 이 128가지 후보 알파벳 조합을 모두 색인과 비교하여 일치하는 단어에 대해서만 처리하는 것도

본 발명의 범주에 포함된다. 그러나 시스템이 입력초기부터 입력값을 인식할 수 있는 시스템에서는 입력값의 증가에 따라 색인에 존재하지 않는 알파벳 조합(위 예에서, 'aa~', 'bdd~', behg~' 등등) 을 입력초기에 제외해 나가는 방법이 유용할 것이다.

<0> 이렇게 반복선택방법에서의 동일한 입력값이 연속하여 증가함에 따라 발생하는 많은 수의 후보 알파벳조합 중에서, 색인에 존재하는 일부의 후보 알파벳조합만을 대상으로 시스템이 검색해 나갈 수 있다. 입력값에 의한 후보 알파벳조합이 모두 색인에 존재하지 않는 것이 확인되면, 시스템은 입력값을 약속된 유형의 심플코드로 간주하여 처리할 수 있다.

91> 이상에서 중국어의 예를 들어 설명하였으나, 중국어 이외의 경우에도 적용될 수 있음은 자명하다. 또한 도1-1의 키패드 뿐만 아니라 도10-*의 키패드 및 기타의 다른 키패드에서도 마찬가지로 적용될 수 있다.

92> 18. 입력값을 동시에 여러가지 유형의 코드(입력값)으로 처리하는 병행입력방법

93> 입력값에 대하여 소정의 어떤 유형들의 코드(입력값)으로 간주하여 처리하는 병행입력방법이 가능하다. 예를 들어 도10-14에서 입력값에 대하여 동시에 풀코드(A)와 전체연관심플코드(B)로 간주하여 처리하면, 시스템은 입력값 "1*#"에 대하여 풀코드로 해석하였을 때의 "bai" 와 전체연관심플코드로 해석하였을 때의 "bei"를 인식해낼 수 있다. 이 경우 시스템은 동시에 "bai"와 "bei"를 사용자에게 제공할 수 있다.

594> 풀우선병행입력방법의 경우 풀코드로 해석하였을 때의 "bei"만을 출력하는 점, 그리고 단축우선병행입력방법의 경우 약속된 단축코드로 해석한 결과인 "bai"만을 출력하는 점이 차이이다. 이러한 병행입력방법에서도 입력값이 특정 유형의

코드에 더 이상 존재하지 않는 것이 확인되는 순간 입력값을 다른 유형의 코드로만 간주하여 처리하는 것이 가능하며, 결국 이는 이미 언급한 병행입력방법의 범주에 포함된다고 할 것이다. 시스템의 색인에 반드시 도10-14에서 처럼 풀코드와 전체연관심플코드를 저장되어 있어야만 하는 것이 아니고, 입력값에 대하여 동시에 여러가지 유형의 코드로 해석하는 경우에 동일하게 적용할 수 있다.

95> 19. 모호성이 발생하는 단어의 우선순위

96> 특정한 어느 한가지 입력방법(예. 특정한 풀입력방법, 혹은 특정한 유형의 단축입력방법)에서, 같은 입력값에 대하여 모호성이 발생하는 경우는 도10-14 및 기타의 도면에서와 같이 특정 입력방법을 적용시 모호성이 발생하는 경우의 우선순위를 근거로 하여 그 순서에 따라 타겟어구를 출력할 수 있다. 예를 들어, 도10-14에서 풀입력방법을 적용시 모호성이 발생하는 경우, 풀코드(A)간의 우선순위인 "(A) 우선순위" 에 따라 타겟어구를 출력하여 사용자에게 추천하여 줄 수 있다.

97> 다음으로 입력값이 소정의 여러 유형의 코드로 해석되는 경우를 살펴본다. 앞서 예시한 대로 도10-14에서 입력값 "1*#" 에 대하여 풀코드로 해석하면 "bai"가 되고, 전체연관심플코드로 해석하면 "bei"가 되는데, 이렇게 서로 다른 유형의 코드로 해석하여 모호성이 발생할 때, 시스템은 풀코드(도면10-14에서 (A))와 전체연관심플코드(도면10-14에서 (B))로 동시에 해석할 때의 우선순위에 의하여 타겟어구를 사용자에게 제공할 수 있다.

598> 예를 들어 입력값을 1차적으로 풀코드로 간주하여 처리하되, 풀코드 생성규

칙을 위반하거나 특정언어의 음절생성규칙을 위반하거나 혹은 색인에 대응하는 풀코드가 더 이상 존재하지 않는 것이 확인 되는 순간, 입력값을 음절기준이니셜코드 및 음절기준이니셜값 심플코드로 간주하여 처리하는 병행입력방법에서, 입력값을 단축코드로 간주하여 처리할 때, "음절기준이니셜코드" 와 "음절기준이니셜값 심플코드" 간의 모호성이 발생할 때의 우선순위에 의거하여 처리할 수 있다. (즉, 음절기준이니셜코드가 (C)이고 음절기준이니셜값 심플코드가 (D) 라면, "(C)+(D)우선순위"에 의하여 타겟어구를 출력하는 것)

)99> 중국어를 예로 들어 설명하였으나, 한자를 입력하는 것이 아니고 병음을 타겟어구로 하면 타 언어의 경우에도 동일하게 적용할 수 있는 것은 자명하다.

)00> 20. 컨트롤처리방법을 풀입력방법으로 하는 병행입력방법

)01> 도2-1, 도2-2와 같이 컨트롤버튼을 [*] 및 [#] 버튼으로 하고 대표알파벳제외 컨트롤처리방법을 적용하는 경우, 병행입력방법을 적용하되 색인에 대표알파벳만으로 이루어진 단어를 모두 저장하고 단축우선병행입력방법을 적용하면, 입력값으로 "*" 혹은 "#"이 눌러지는 순간 시스템은 입력값을 풀코드로 간주하여 처리할 수 있음을 지적하였다.

)02> 도2-1 및 도2-2의 일본어의 경우 단축코드는 모두 숫자값으로만 이루어져 있다고 가정한다. 이러한 경우에 대표알파벳만으로 이루어진 모든 단어를 저장하고 있는 색인(단축코드색인에 포함되어 구성될 수도 있고 단축코드 색인과 별도로 구성될 수도 있음)을 두고, 병행입력방법을 적용하되 입력값으로 숫자버튼만이 계속 눌러지는 동안은 입력값을 풀코드로 해석한 결과와 대표알파벳만으로 이루어진 단어의 색인(혹은 이를 포함하고 있는 심플코드색인)을 참조하

여 단축코드로 해석한 결과를 모두 소정의 우선순위에 의하여 타겟어구로 사용자에게 제공할 수 있다.

3> 예를 들어, 입력값 도2-1, 도2-2에서 입력값 "111" 에대하여 풀코드로 처리하면(즉, 풀우선병행입력방법을 적용하면), "ああ" 로 해석되는데, 만약 대표알파벳만으로 이루어진 모든 단어의 색인을 참조하여 "あ~" 로 시작하는 단어가 존재하지 않고, "あい" 혹은 "あい~" 만이 존재한다면, "あい" 또는 "あい~" 를 "ああ~" 보다 우선하여 사용자에게 제공할 수 있는 것이다.

4> 도10-15, 도10-16을 참고한다. 도10-15에서 입력값 "11" 에 대하여 풀코드로 해석하되, 이것이 대표알파벳만으로 이루어진 단어이므로 다시 "대표알파벳만으로 이루어진 모든 단어의 색인" 을 검색하여 유효한 단어인지 확인한다. 여기서 "11" 을 풀코드로 해석한 "ああ" 가 색인에 존재하면 입력값에 대하여 이를 유효한 단어로 인식하면 된다. 대표알파벳으로만 이루어진 모든 단어의 색인에 입력값에 대응하는 어구가 더 이상 존재하지 않는 것이 확인되는 순간 시스템은 입력값을 다시 심플코드로 해석하여 이에 대응하는 "あい" 를 시스템이 인식하게 된다. 도10-15에서 시스템은 입력값 "11" 대하여 최우선으로 "あい" 로 인식하게 된다. "11" 다음의 입력값에 따라 "あい..." ("..." 은 "あい" 다음에 다른 알파벳이 덧붙어 있는 경우) 로 될 수 있으므로, 시스템이 이를 "あい" 다음 순위로 출력할 수도 있다. 마찬가지로 색인에 존재하지 않더라도 타겟어구가 "ああ" 일 수도 있으므로 이 역시 "あい" 의 다음 순위로 출력할 수도 있다. 도10-15에서 "あい..." 과 "ああ" 가 흐린 색으로 그리고 2번째와 3번째 추천순서로 표시된 것은 시스템이 이렇게 출력할 수도 있고 단지 "あい" 만을 출력할 수 있음을 나타낸다.

5> 도10-15에서 (1), (2), (3)은 순차적으로 일어날 수도 있고, "입력값을 동시에 여러 유형의 코드로 인식하는 병행입력방법" 에서 언급한 것처럼 동시에 일어날 수도 있다. 마찬가지로 색인에 풀코드 및 심플코드를 표시한 것은 이해를 돕기 위한 것일 뿐이고, 풀코드 및 심플코드가 저장되어 있지 않아도, 동일한 절차를 수행할 수 있음은 자명하다 (심플코드가 음절기준이니셜코드 혹은 전체연관심플크드와 같이 규칙적으로 해석될 수 있는 심플코드인 경우). 도10-16에서와 같이 대표알파벳으로만 이루어진 모든 단어의 색인과 심플코드 색인이 하나의 색인으로 존재하는 경우도 동일한 내용이다. 구현상의 자료구조는 다양한 형태로 실현될 수 있다.

06> 시스템의 작동 결과는 기 설명한 병행입력방법과 유사하나, 단축우선병행입력방법을 적용하는 것이 아니고, 풀우선병행입력방법 혹은 일반적인 병행입력방법을 적용하는 것으로 볼 수 있다.

07> 이는 대표알파벳제외 컨트롤처리방법을 풀입력방법으로 하는 병행입력방법에서 뿐만 아니라 컨트롤처리방법을 풀입력방법으로 하는 모든 병행입력방법에서 동일하게 적용될 수 있다.

08> 17. 기능의 입력

09> 기능 중 어구(문장)을 입력시 공백은 필수기능이라 할 수 있고, 삭제(입력취소)기능 역시 필요하다. 엔터기능은 입력후 일정시간 동안 무입력으로 해결할 수 있으나 역시 필요한 기능이라 할 수 있다. 이 3가지 기능을 통칭하여 편의상 "3기본기능" 이라고 부른다. 또한 영어와 같이 대소문자가 존재하는 언어에 있어서는 대/소문자전환 기능도 필요하다. 상/하/좌/

우 이동기능 중 좌/우 이동기능은 삭제 그리고 공백기능으로 대체할 수 있고 상/하 이동기능은 필수기능은 아니라고 볼 수 있다.

<0> 여기서 어구 입력을 위한 각종 기능 중 필수기능 순으로 나열해 본 사례를 몇가지 들면 다음과 같다. 물론 이는 참고사항일 뿐이고, 절대적인 기준은 아니며, 특정한 상황에 따라 조금씩 달라질 수 있다. 만약 영어 키패드를 통하여 중국어를 입력하는 경우는 한자변환 기능이 가장 필수기능이 될 것이기 때문이다. 또한 일본어나 한국어와 같은 한자 문화권 언어에 있어서도 한자변환 기능은 중요한 기능이 될 수 있다.

<1> 1. 공백 - 엔터 - 삭제 - 대/소문자 전환 - 자국어/숫자/영어 모드전환 - 이동(상/하/좌/우)

<2> 2. 공백 - 삭제 - 엔터 - 자국어/숫자/영어 모드전환 - 대/소문자 전환 - 이동(상/하/좌/우)

<3> 3. 공백 - 엔터 - 삭제 - 한자변환 - 자국어/숫자/영어 전환 - 이동(상/하/좌/우)

<4> 4. 한자변환 - 공백 - 엔터 - 삭제 - 영어/숫자 모드전환 - 이동

<5> 17.1 컨트롤처리방법에 의한 기능의 입력

<6> 알파벳 및 각종 기호를 컨트롤 처리방법에 의하여 선택할 수 있는 것을 설명하였다. 각종 기능을 선택하는데에도 역시 동일하게 적용될 수 있다. 도12-1은 숫자 버튼에 각종 기능의 의미를 부여하고, 이를 "기능컨트롤"과 조합하여 구현할 수 있음을 보여주는 사례이다. 각 버튼에 연관된 숫자버튼을 쉽게 기억할 수 있도록 연관지어 지정할 수 있다. 도12-1에서 좌우상하 화살표는 직관적으로 인지할 수 있으며, 중앙에 엔터를 두는 것 역시 용이하게 숫자버튼

과 그에 연관된 기능을 알 수 있게 하기 위함이다. 나머지 필요한 기능역시, 각 언어별로 기능의 명칭과 관련있는 알파벳이 배치된 버튼에 연관지음으로써, 기억을 용이하게 할 수 있다. 도12-1에서 shift 기능을 "S" 가 배정된 [7] 버튼과 연관하는 것이다.

7> 도12-1에서 편의상 선출원의 한국어의 경우에서 처럼 [*] 버튼을 누름횟수에 따라 격음컨트롤, 경음컨트롤이 선택되고 그 다음으로 기능컨트롤이 선택된다고 가정할 때, 엔터 = [5]+{기능} = [5]+[*]+[*]+[*] 이 된다. 만약 [5] 버튼1타로 선택하는 알파벳이 격음/경음이 존재하지 않는 알파벳이면, 건너뛰기 컨트롤처리를 적용하여 [5]+[*] 까지 눌렀을 때 엔터가 입력된 것으로 할 수도 있다.

18> 각 버튼에 연관된 기능은 숫자버튼위에 표시될 수도 있고 표시되지 않을 수도 있다. 그런데 각종 기호의 그룹과 연관된 숫자버튼을 쉽게 알 수 있도록 액정화면의 일부분에 표시하여 줌으로써, 버튼위에 표시하지 않고, 간결한 배치를 유지하면서, 사용의 편리를 더할 수 있음을 지적하였다. 이는 여기서도 동일하게 적용될 수 있다. 도12-2를 참고한다. 또한 기호입력에서 지적한 바와 같이 기능과 숫자버튼과의 연관되는 기능을 사용자가 재정의하도록 할 수도 있다.

19> 영어의 경우와 같이 [*] 혹은 [#] 버튼이 다른 컨트롤버튼으로 사용되지 않는 경우는 [*] 버튼 1타로 "기능컨트롤" 을 선택할 수 있으므로, 도12-1에서 기능이 연관된 숫자버튼과 [*] 버튼을 한번 눌러 기능을 입력할 수 있다.

20> 17.2 다차원 교차컨트롤처리방법에 의한 기능의 입력 (컨트롤버튼의 교차 조합 이용)

1> 하나의 알파벳을 입력하기 위하여 한가지의 컨트롤버튼이 사용되는 경우를 1차원 컨트롤처리방법이라고 하였다. 같은 컨트롤버튼이 반복적으로 사용되는 경우도 마찬가지이다.

2> 컨트롤버튼으로 [*], [#]이 사용되고, 1차원 컨트롤처리방법으로 알파벳 및 각종의 기호를 입력하는 경우, 컨트롤버튼의 교차 조합(이 경우 [*]+[#], [#]+[*])을 이용하여 각종 기능을 입력할 수 있다. 컨트롤버튼이 반복적으로 사용되지 않는 경우라면, [*]+[*] 혹은 [#]+[#] 등을 이용하여 기능을 입력할 수도 있다. "[*]+[#]" 는 왼쪽에서 오른쪽으로 진행하는 모양이므로 우측이동 기능(혹은 공백)으로 활용시 사용자가 쉽게 인지할 수 있다. 반대로 [#]+[*] 는 왼쪽에서 오른쪽으로 좌측이동기능(혹은 삭제)으로 활용시 사용자가 쉽게 인지할 수 있다.

23> 이를 다차원 교차 컨트롤처리방법의 관점에서 이해할 수 있다. 입력예로 컨트롤 후입력을 적용하여, ^a = a + [*] 로 입력하는 경우, 사용자가 "a + [*] +[#]" 을 입력하면, a + [*] 까지 입력시 시스템은 ^a 이 입력된 것으로 인식하나, 다음에 [#] 이 올 수 없으므로([#] 버튼이 다른 컨트롤버튼으로 사용되고 있더라도), [#] 이 입력되는 순간, 시스템은 "[*] +[#] " 이 우측이동기능을 입력한 것으로 인식할 수 있다. 컨트롤선입력으로 지정한 경우도 유사함을 알 수 있다.

24> 일본어의 경우에 2차원컨트롤처리방법이 적용되었다. 이 경우 컨트롤버튼을 3회(혹은 그 이상) 번갈아 조합함으로써 기능을 구현할 수 있다. 예를 들어, 공백 = [*]+[#]+[*], 삭제 = [#]+[*]+[#] 로 입력하는 것이다. 공백 기능의 사례에서 [*]+[#]+[*] 을 눌렀을 때 형성되는 모양이 마치 오른쪽 화살표의 화살촉을 연상시키므로 이를 쉽게 기억할 수 있다. 삭제의 경우도 마찬가지로 [#]+[*]+[#] 를 눌렀을 때 왼쪽 화살표의 화살촉을 연상시킬 수 있으므로 이를 쉽게 기억할 수 있다. 이것이 가능한 이유는 앞에서 설명한 것과 동일하다.

5> 이를 정리하면, 1차원컨트롤처리방법을 적용하는 경우 컨트롤버튼을 2회번갈아 조합하고, 2차원컨트롤처리방법을 적용하는 경우는 컨트롤버튼을 3회 번갈아 조합함으로써 각종기능을 입력할 수 있는 것을 알 수 있다. 즉 알파벳 및 각종의 기호의 입력을 위해 N차원 컨트롤처리방법이 적용되는 경우, N+1회 컨트롤버튼을 번갈아 조합함으로써, 각종 기능을 입력할 수 있는 것이다.

6> 18. 한자 변환

7> 한국어, 중국어 한어병음과 한자는 1:多 의 대응관계 이다. 예를 들어 "예의" 에 대응되는 한자는 禮意, 禮儀, 銳意 … 와 같이 다수 있게 된다.

8> 중국어의 경우도 "beijing" 에 대응되는 한자는 北京, 背景 등이 있게 된다(성조부호를 붙이지 않았을 경우). 따라서 한자변환버튼을 반복적으로 눌러 후속 한자(예. 北京(2nd), 背景(3rd)) 를 선택하도록 할 수 있다. 예를 들어 예의를 beijing 을 입력후 한자변환기능을 한번누르면 北京 이 되고 한번 더 누르면, 背景 이 되는 것이다. 혹은 리스트에서 선택하도록 할 수 있다.

29> 중국어의 경우는 한자를 입력하기 위하여 입력을 원하는 모든 한자(편의상 "타겟한자"라 부름)를 색인에 저장하고 있어야 하므로, 한어병음을 입력하는 중간에라도, 타겟한자를 확정할 수 있으면, 이를 사용자에게 제공할 수 있다. 예를 들어 "beij …" 으로 시작하는 단어가 "北京" 밖에 없다면, 사용자가 "beij" 까지 입력시 "北京" 을 사용자에게 제공할 수 있는 것이다. 만약 병음색인에 "beijing(北京)"이 단 하나만 존재한다면, 입력 중간에 사용자가

타겟 한자를 확정하지 않더라도, "beijing"이 모두 확정적으로 인식되는 순간 시스템은 "beijing"을 "北京"으로 변환하여 출력할 수 있는 것이다.

<0> 그리고 일본어의 경우는 히라가나와 가타가나는 1:1의 대응관계를 가지고, 히라가나 혹은 가타가나와 한자는 1:多의 관계를 가진다. 따라서 히라가나 입력후 가타가나 변환컨트롤(예. 도2-1에서 "아/아" 컨트롤)을 입력하여 가타가나로 변환시키는 것이 가능함을 설명하였다. 만약 도2-1에서 변형알파벳을 교차컨트롤처리방법에 의하여 입력하고, "아/아" 컨트롤을 [0] 버튼 2타로 선택할 수 있다. 여기서 다시 특정 히라가나 혹은 가타가나에 대응되는 한자들을 [0] 버튼의 반복누름으로 선택하도록 할 수 있다. 예를 들어 "아/아" 컨트롤을 후입력하는 것으로 지정하였다면, 히라가나 단어(혹은 알파벳)를 입력후 [0]+[0] 을 누르면 입력한 히라가나에 대응되는 가타가나로 변환되고, 다시 [0] 버튼을 누르면, 첫번째 한자로 변환되고, 다시 [0] 버튼을 한번 누르면 그 다음 한자로 변환되는 것이다. 반대로 가타가나 입력모드에서는 가타가나 입력후 [0]+[0] 을 누르면 대응되는 히라가나로 변환되고, 다시 [0] 버튼을 한번 누르면 한자로 변환되도록 할 수 있다.

<31> 19. 심플코드 색인의 구축

<32> 출원인이 제시한 심플코드 중 전체연관심플코드, 음절기준이니셜코드, 자음연관심플코드, 첫모음+자음연관심플코드, 단어기준이니셜코드 등은 기계적으로(즉, 자동으로) 특정 어구(단어 혹은 구절)로부터 심플코드를 추출해낼 수 있는 좋은 성질이 있다. 따라서 사용자가 풀입력방법을 사용하여 한번 입력한 단어에 대하여 심플코드 색인(심플코드 및 심플코드 대응어구로 이루어진 색인)을 자동으로 구축하고, 이를 병행입력방법에 활용할 수 있다. 심플코드가 저

장되는 것은 도5-5와 유사한 환경에서 클라이언트측 혹은 서버측이 될 수도 있고, 혹은 클라이언트측과 서버측 모두에 저장될 수도 있다.

<33> 같은 어구에 대하여 다른 종류의 심플코드(예. 전체연관심플코드 및 음절기준이니셜코드)가 생성되도록 할 수도 있다. 그러나 일반적으로 특정유형의 심플코드를 사용하는 것이 편리할 것이며, 사용자는 자동으로 생성되는 심플코드의 종류를 인지하고 있으면 되며, 이를 병행입력방법 적용시에 사용할 수 있다. 이를 적용하면, 사용자의 문자입력이 많아질수록 심플코드 색인이 풍부해지는 효과가 있다.

<34> 특정 언어의 특정 어구로부터 음절기준이니셜코드를 추출해내기 위해서는 특정 단어로부터 음절을 분리해 내야 한다. 특정 언어의 특정 단어에서 음절을 분리해 내는 것은 특정 언어의 분철(음절분리)규칙에 따르면 가능하다. 각 언어마다 분철규칙을 가지고 있으므로 이를 이용하여 특정 단어로부터 음절을 자동으로 분리해내는 것은 어려운 일이 아니며 여기서는 자세히 언급하지 않는다.

<35> 예를 들어, 중국어 병음에서 "Zhongguo" 의 경우, 모음 "o" 와 "uo"를 중심으로 "Zhong" 와 "guo" 로 음절분리할 수 있으며, 이는 본 발명에서의 병음을 인식하는 과정보다 훨씬 더 쉽게 이루어질 수 있다. 단어의 앞에서부터 forward scanning 하여 음절을 분리해내는 예를 보이면, "zho"다음 자음 "n"이 다음 음절의 자음인지 알 수 없지만, 그 다음 자음 "g"를 보아 첫 음절에 속한 자음임을 알 수 있고, 마찬가지로 "g"는 다음 음절의 자음이 될 수도 있지만, 그 다음 "g"를 보아 첫번째 "g"는 첫음절에 속한 자음이고 두번째 "g"는 다음 음절을 이루는 자음임을 알 수 있는 것이다. (병음에서 어느 한 음절이 "~ng"로 끝날 수 있음) "zhongg"까지 스캐닝하여 첫번째 음절 "zhong"를 분리해 낼 수 있다. 다음으로 "u"는 당연히 두번째 음절을 이루는 모음이고, 연속하여 나오는 모음 "o" 역시 두번째 음절을 이루는 모음임을 시스템이 인

식할 수 있다. (병음에서 영어모음 "uo"가 연속하여 나올 수 있음) "o" 다음에 나오는 알파벳이 더이상 없으므로 시스템은 "zhongguo" 의 음절을 "zhong" 와 "guo"로 분리해 낼 수 있고, 따라서 음절의 이니셜값이 "zg"인 것을 인식할 수 있으며, "zhongguo"의 음절기준 이니셜코드가 "43" (도면 10-6기준) 인 것을 식별해낼 수 있다. 나머지 언어의 경우에도 해당 언어의 음절분리규칙에 따라 특정 단어 혹은 구절로부터 자동으로 음절을 분리해 낼 수 있는 것은 물론이다.

36> 20. 특수부호(첨자)를 붙이지 않은 어구로부터의 타겟어구 해석

37> beijing 의 경우, 모음에 성조부호를 붙이지 않는 경우 "北京" 혹은 "背景" 에 모두 대응된다고 하였고 이 경우 특정버튼을 반복적으로 눌러 타겟어구를 선택하거나, 리스트에서 양자중 하나를 선택할 수 있도록 할 수 있다고 하였다.

38> 마찬가지로 중국어 이외의 언어에서도 성조부호를 붙이지 않거나, 첨자가 붙은 모음을 입력하여야 함에도 기본모음만을 입력한 경우, 특정버튼의 반복누름에 응답하여 색인을 검색하여 타겟어구를 사용자에게 제공하거나, 색인을 검색하여 첨자가 붙은 모음을 포함하는 대응어구의 리스트를 사용자에게 제공하여 사용자로 하여금 선택하도록 할 수 있다.

39> 이는 베트남어와 같이 변형알파벳이 다수 있고, 성조부호등이 복잡하게 구성된 언어에서 유용할 것이다. 단 기본알파벳에 첨자가 붙어 이루어진 어구의 색인을 시스템에서 구비하고 있어야 한다. 마찬가지로 이러한 색인을 구축하는 것도 사용자가 기 입력한 단어로부터 구축할 수 있다. 예를 들어 문자입력을 통하여 사용자가 "^+abc" 라는 단어('a' 위에 첨자 '^' 붙은 경우)를 입력하면, 시스템은 "^+abc" 을 시스템 저장하고 있다가(시스템에서 "

abc" 와 'abc' 에 대응되는 "^+abc" 를 모두 저장하고 있어도 무방함), 다시 사용자가 "abc" 입력후 특정버튼을 누르면 "^+abc" 를 사용자에게 제공하거나 사용자가 "abc" 를 입력시 "^+abc" 와 같이 "abc" 에 대응하는 어구들의 목록을 사용자에게 제공할 수 있다. 만약 "abc" 가 타겟어구라면 사용자는 특정버튼을 누르지 않거나, 시스템이 관련어구의 리스트을 제공하더라도 'abc' 를 선택하게 될 것이다. 리스트가 제공되는 시점은 단어의 입력이 종료되는 시점이 될 수 있고 입력중간의 임의의 시점이 될 수 있다.

10> 25. 병행입력방법에서의 보완 사항

11> 단어의 입력(풀코드 입력)에 따라 입력되는 내용과 일치하는 단어를 색인으로부터 출력하여 주고, 사용자로 하여금 선택하도록 하는 기법은 널리 사용되고 있었다. 예를 들어 윈도우의 "자동완성기능" 을 생각하면 된다. 풀입력(풀코드에 의한 입력)시에도 색인(예. 병행입력을 위한 색인)을 참조하여 색인에 저장된 어구(단어 혹은 구절) 중에서 일치하는 어구를 출력하여 주고 사용자로 하여금 선택/확정하도록 할 수 있음은 물론이다.

12> 이를 도4-5 ~ 도4-8의 입력시스템을 기준으로 설명하면 다음과 같다. 예를 들어 색인에 "고맙습니다" 와 "고마워요" 등의 단어가 저장되어 있는 것으로 가정한다. "10* = 고" 까지 입력되면, 시스템은 "고..." 와 일치하는 단어(즉, "고" 로 시작하는 단어로써, "고맙습니다", "고마워요", . . .)를 적절한 형태로 출력하여 줄 수 있다. 휴대폰과 유사한 환경에서는 입력값에 의하여 생성되는 내용은 입력라인(도13-1에서 액정 상단)에 보어지게 되고, 색인을 참조하여 출력하

는 후보 어구(단어 혹은 구절)들은 하단에 목록 혹은 목록과 유사한 형태로 출력될 수 있다. 사용자는 이동버튼(네비게이션 버튼) 등을 이용하여 원하는 단어를 선택한 후 확정하여 입력할 수 있다. 휴대폰 액정을 예로 들어 도13-1을 참고한다. 도13-1은 풀입력에 따라 형성되는 음절과 일치하는 단어를 액정의 하단에 출력하는 사례이다.

43> 여기서 "고맙습니다" 를 입력시, "고" 까지 입력한 후 나머지 음절을 단축입력(예시에서 음절기준 이니셜값 심플코드를 이용한 단축입력)으로도 입력할 수 있다. 즉 "고+ㅁㅅㄴㄷ = 10*+5723" 으로 입력하는 것이다. 여기서 첫번째 음절이 "고" 이고 나머지 음절은 이니셜값이 "ㅁㅅㄴㄷ" 에 대응되는 단어 (즉, "고맙습니다")를 사용자에게 제공할 수 있다.

44> 단, 한국어의 음절 중 약 절반은 종성(받침)이 있는 음절이고, 나머지 절반은 종성이 없는 음절이다. "고맙습니다" 의 경우 "1*05723" 를 연속하여 입력할 경우 "곰ㅅㄴㄷ" 로 되는데, 시스템은 "곰ㅅ = 1*057" 까지는 정상적인 풀입력으로 처리하다가 "2" 가 입력되는 순간, 입력값의 일부가 정상적인 한국어 음절을 이루지 못하는 것을 인식하고 나머지 음절을 이루지 못하는 부분(예. "ㅅㄴㄷ = 723")을 단축입력으로 간주하여 처리할 수 있다. 이 역시 선출원에서 언급한 "병행입력방법 (풀(입력)우선 병행입력방법)" 의 범주에 속하는 것으로, 풀입력에 의하여 인식된 음절(예. "곰")은 당연히 풀코드로 해석하고, 나머지 입력값(예. "ㅅㄴㄷ = 723")은 심플코드로 해석하는 것이다.

45> 여기서 풀입력에 의하여 기 형성된 음절을 포함하여 색인에서 일치하는

단어(예. 색인에 "곰습니다" 가 저장되어 있을 경우)를 찾아 제공할 수도 있고, 심플코드로 해석하는 입력값에 대응되는 단어(예. 색인에 "습니다" 가 저장되어 있을 경우)를 제공할 수도 있고, 양자의 경우를 모두 출력할 수도 있다. 양자를 모두 출력시에는 정해진 약속(예. 풀코드로 해석된 부분을 포함하는 단어를 우선, 혹은 반대로, 혹은 단어의 우선순위에 의거, 혹은 특정 그룹의 단어들에 대하여 우선)에 의하여 특정 단어가 상위에 출력되도록 할 수 있다. 다양한 수단이 가능하므로, 이러한 출력 순서를 사용자가 설정할 수 있도록 하는 것이 바람직할 것이다.

<16> 양자를 모두 출력하는 경우 "곰ㅅㄴㄷ" 에 대응하여, "곰습니다" 와 "습니다" 가 액정 하단에 출력되었을 때, 사용자가 "곰습니다" 를 선택하면, "곰ㅅㄴㄷ" 전체가 "곰습니다" 로 교체될 것이며, "습니다" 가 선택되면 이에 대응되는 "ㅅㄴㄷ" 이 "습니다" 로 교체되어 "곰습니다" 가 되는 것은 당연하다.

<17> 여기서 색인에 "고맙습니다" 만이 저장되어 있다면, 입력값 "1*05723" 의 연속입력에 대응하여 "고맙습니다" 를 출력하여 줄 수 없게 된다. 따라서 시스템은 추가적으로 "1*05723" 에 대하여 해석이 가능한 "곰ㅅㄴㄷ" 와 "고ㅁㅅㄴㄷ" 로 동시에 해석하여 색인을 참조하여 일치하는 단어를 출력(사용자에 제공)할 수 있다. "곰ㅅㄴㄷ" 을 "고ㅁㅅㄴㄷ" 로 해석하는 것은 단축입력의 시작으로 인지된 "ㅅ" 앞의 음절이 종성자음으로 끝난 경우, 이를 앞자음에서 제외하고(즉, "고"), 종성자음을 낱자음으로 "고" 와 "ㅅ" 사이에 넣으면 된다. 조금 복잡해 보이나 한글처리에 대하여 기본적인 지식이 있는 사람은 쉽게 알 수 있다. 이중자음이 종성으로 사용된 경우도 유사하게 적용할 수 있다. 도13-2는 출력의 우선순위를 "우선출력하는 것으로 지정된 특정 그룹의 단어(들) ? 풀코드로 해석된 부분을 포함하는 단어(들) ? 풀코드로 해석된 부분을 포함하지 않는 단어(들) ? . . ." 의 순서를 적용하여 후보 단어를

출력한 사례를 보여준다. 여기서 "습니다" 는 우선출력하는 것으로 지정된 그룹에 속한 것으로 하여 가장 상위에 출력되었다.

<8> 만약 사용자가 "고ㅁㅅㄴㄷ" 로만 해석되도록 하고 싶다면, "10* = 고" 까지 입력 후 음절확정을 줄 수 있는 약속된 조작(예. 약속된 일정시간 무입력, 혹은 오른쪽 화살표 [>]버튼 한번 누름 등)후 "5723" 을 누름으로써 "고ㅁㅅㄴㄷ" 로만 해석되어 이에 대응하는 단어만을 검색하도록 할 수 있다. 이 경우에도 기 인식된 음절인 "고" 를 포함하여 "고ㅁㅅㄴㄷ" 에 대응되는 단어(예. "고맙습니다") 를 출력할 수도 있고, "고" 를 제외하고, "ㅁㅅㄴㄷ" 에 대응되는 단어(예. "맞습니다" 가 색인에 저장되어 있을 경우)를 출력할 수도 있고, 양자의 경우를 모두 출력할 수도 있다. 도13-3을 참고한다.

<9> 여기서 의도적인 음절확정을 위한 조작 버튼으로 오른쪽 화살표 버튼(이하에서 "[>]버튼" 으로 표기)을 한번 누르는 것이 음절확정으로 작용하도록 할 수 있다. 더불어 [>]버튼은 공백버튼으로도 활용될 수 있는데, 음절확정이 이루어진 다음에는 공백의 입력으로 작용할 수 있다. 즉 "고" 입력후 한번 [>]버튼이 눌러지면, 음절 "고" 가 확정되고, 한번 더 [>]버튼이 눌러지면, "고 " ('고' +공백)으로 되는 것이다. 즉 "10*>5723" 이 입력되면, 시스템은 이를 "고ㅁㅅㄴㄷ" 로 해석하여 이에 대응되는 단어인 "고맙습니다" 를 사용자에게 제공할 수 있는 것이다. 물론 시스템이 예시한 색인을 참조하는 경우, "10*>57 = 고ㅁㅅ" 까지만 입력되더라도 시스템은 "고맙습니다" 를 출력할 수 있음은 자명하다.

<50> 이러한 병행입력방법에서, 풀입력에 의하여 형성된 음절을 포함하여 처리하는 경우는 풀입력 중간에 일치하는 후보단어를 제공시 후보단어의 목록에서 타겟단어를 확정하도록 하는 경우에 유용할 것이다. 또한 기 형성된 음절을 제외하고 단축입력으로 처리되는 입력값(즉, 심플코드로 해석하는 입력값)에만 대응되는 후보단어를 출력하는 경우는 자주사용되는 접미어 혹

은 어조사 등을 단축입력으로 신속히 입력하는데 유용할 것이다. 예를 들어 "고맙" 까지 입력후, (소정의 수단으로 입력내용을 확정하고), "ㅅㄴㄷ" 로 "습니다 (색인에 저장되어 있는 것으로 가정시)" 를 입력하는 것이다. 이렇게 "...습니다" 와 같이 자주 사용되는 접미어는 색인에서 별도의 그룹(예. 접미어 그룹)으로 두고 병행입력에서의 후보단어 출력시 해당 그룹의 단어가 우선적으로 출력되도록 할 수 있다. 만약 "고맙ㅅㄴㄷ" 의 입력시, 색인에 "고맙습니다" 와 "습니다" 가 모두 저장되어 있다면, 후보 단어목록에 "고맙습니다" 와 "습니다" 가 모두 출력될 수도 있다.

51> 한가지 더 예를 들면, "세종ㄷㅇ = 70##90*838" 입력시 "세종" 까지는 풀입력으로 처리하고, 적절한 음절을 이루지 못하는 "38" 을 단축입력값으로 처리할 수 있다. 마찬가지로 색인에 "세종대왕" 만이 있으면, "세종대왕" 이 출력될 것이다. 반대로 색인에 "대왕" 만이 있으면, "대왕" 만이 출력될 것이다. "세종대왕" 과 "대왕" 이 모두 있으면, 모두 출력될 수도 있으나, 약속에 의하여 한가지(풀입력으로 처리한 부분을 포함/불포함)만 출력되도록 하는 것이 바람직할 것이다. 또, "문무대왕" 의 경우 "문무" 까지 풀입력으로 입력후, "ㄷㅇ" 을 연속하여 입력하면, "문묻ㅇ" 이 될 것이다. 따라서 "문무" 까지 풀입력으로 입력하고, "대왕" 만을 단축입력으로 입력하려면, "문무" 입력후, 입력된 음절 "문무" 확정(예. 음절의 종료를 줄 수 있는 오른쪽 화살표 [>]버튼을 한번 눌러주어 확정) 후, "38 = ㄷㅇ" 을 입력할 수 있다. 여기서 "대왕" 이 "ㄷㅇ = 38" 에 대응되는 색인의 다른 단어(예. "다와") 보다 우선순위가 낮더라도, 색인에 "문무대왕" 이 저장되어 있다면 입력값 "5*025*0>38" 에 대하여 다른 후보단어(예. "다와") 보다 우선하여 "대왕" 을 출력하도록 할 수 있다.

52> 더 나아가 "ㄱㅁ습ㄴㄷ = 157*623" 와 같이 단축입력으로 입력하다가 일부의 음절을 풀입력에 의하여 입력하는 경우도, 이에 대응되는 "고맙습니다" 로 처리하는 것이 가능하다. "157 = ㄱㅁㅅ" 까지 입력되면 시스템은 심플코드(예시에서 음절기준 이니셜값 심플코드 혹은 음절기준 이니셜코드) "157" 를 해석하여 각 음절이 "ㄱㅁㅅ" 에 대응되는 단어를 검색한다. 다시 "*6" 이 입력되면, "157*6 = ㄱㅁ습" 으로 "7*6 = 습" 이라는 음절이 형성되는 것을 시스템이 인식할 수 있다. 즉 기 입력된 "15" 가 단축입력값임을 시스템이 인지하고, 다시 "23" 이 입력되면, 이는 적절한 음절이 형성되지 않으므로 단축입력값으로 처리할 수 있다. 역시 이 사례에서도 시스템은 "ㄱㅁ습ㄴㄷ" 에서 일부음절은 풀코드로 해석하고, 또 일부음절은 단축코드로 해석하여 이에 대응하는 "고맙습니다" 를 사용자에게 제공할 수 있다.

53> 중국어의 경우 한 음절은 "성모+운모" 로 이루어져 있다고 하였다. 한어병음표시시 "성모" 는 로마알파벳의 자음이고, "운모" 는 로마알파벳의 모음 혹은 "모음+n" 혹은 "모음+ng" 로 되어 있다고 하였다. 즉 병음으로 표기되는 중국어 한자 한 글자(한 음절)는 로마알파벳의 "자음+모음" 혹은 "자음+모음+n" 혹은 "자음+모음+ng" 로 이루어져 있는 것을 알 수 있다. 드물게 모음 "a" , "e" , 또는 "o" 로 시작하는 음절이 있을 수 있다. 여기서 병음 한 음절의 끝에 올 수 있는 로마알파벳 자음은 "n" 혹은 "ng" 만이 가능함을 이미 지적하였다. 이는 한국어의 종성처럼 간주될 수 있다. 종성(한 음절의 끝자음)으로 쓰일 수 있는 자음이 "n" 혹은 "ng" 만이 가능한 것이다.

54> 예를 들어 병음색인에 "中華(zhonghua)" , "民國(minguo)" , "中華民國(zhonghuaminguo)" , . . . 등의 단어가 저장되어 있다고 가정하면, "中華(zhonghua =

49***7739##*, 도10-6기준)" 까지 풀입력에 의하여 병음을 입력하고, "民國(minguo)" 만을 단축입력(예시에서 단축입력값으로 음절기준 이니셜값 심플코드를 이용한 단축입력)으로 "mg = 73 (도10-6기준)" 입력하는 예를 든다. "中華" 의 입력에서 해당 병음("zhonghua = 49***7739##*") 입력후 나열된는 후보한자 중 선택하여 확정하는 과정이 매개되고 나서 "mg = 73" 이 입력될 수도 있고, 병음 "zhonghua = 49***7739##*" 입력후 연속하여 "mg = 73" 이 입력될 수도 있다. 전자의 경우는 "73 = mg" 에 대응되는 "民國(minguo)" , 民歌(minge), . . . 등의 단어만을 후보단어로 출력하는 것이 당연하다. 후자의 경우는 입력값 "49***7739##*" 에 대하여 "中華" 로 해석하고, 단축입력값으로 인식된 "73" 에 대응되는 "民國" 과 함께 "中華民國" 을 출력하는 것이 또한 당연하다. 입력값 "49***7739##*73 = zhonghua + mg" 에 대하여 "49***7739##*7" 까지의 입력에 대하여, 마지막의 "7" 은 병음성모의 입력으로 정상적인 풀입력으로 시스템이 인지하나, 마지막의 "3" 이 입력되는 순간 시스템은 최종 입력된 2개의 입력값 "73" 이 단축입력값임을 인지하게 된다. 이는 기 설명한 한국어의 경우와 동일하다. 따라서 시스템은 입력값 "49***7739##*73" 에 대하여 풀입력값 "49***7739##*" 에 대응하는 "中華" 와 단축입력값 "73" 에 대응하는 "民國" 으로 이루어진 단어인 "中華民國" 을 출력할 수 있게 되는 것이다. 도13-4을 참고한다.

355> 반대로 "中華" 를 단축입력에 의하여 "zh = 49" 로 입력후 후보 단어 중 "中華" 를 선택하여 입력하지 않고, 연속으로 "民國" 을 풀입력으로 "7#773##***" 을 입력하였을 때 (즉, "497#773##*** = zhminguo")에도 "z__h__minguo" 에 대응되는 "中華民國" 을 출력할 수 있다. 마찬가지로 "民國 (minguo)" 의 입력을 위하여 "73##***" 입력시 "73##*** = mguo = m__guo" 로 해석하여 "民國" 을 출력할 수 있다. "73" 까지 눌러지는 순간 시스템은 단축입력으로 간주하고 "m__g__" 에 대응되는 단어를 검색하여 처리하게되나, 다음에 오는

"##*** = uo" 를 두번째 음절을 이루는 모음으로 처리하는 것이다. "mg" 다음 "uo" 를 입력하는 것은 "m__g__" 에 대응되는 후보 단어들 중 "民國" 을 압축하여 주는 역할을 한다.

<6> 기타의 수단(단축입력, 풀입력, 획을 이용한 입력 등 모든 한자입력수단)으로 한자 "중화" 를 기 입력하고 나서, "민국" 을 단축입력에 의하여 "73" 으로 입력시, 단순히 입력값 "73" 를 심플코드로 해석하여 "m__g__" 에 대응되는 단어를 사용자에게 제공한다면, 다수의 후보단어들(예. "民國(minguo)" , 民歌(minge), . . .)이 목록의 형태로 출력될 것이다. 그러나, "中華民國" 이 저장되어 있으면, "民國" 이 색인에 저장되어 있는 것과는 별개로, "中華+mg = 中華+73" 을 "中華民國" 으로 해석하는 것이 가능하다. 이 경우 "中華" 는 이미 한자로 입력된 상태이므로, 입력값 "mg = 73" 에 대응하여 "民國" 을 다른 후보단어(예. "民歌(minge)" , . . .)보다 우선하여 출력할 수 있다.

<57> 여기서 "zhong..." 가 입력되었을 때, 시스템은 "zho" + "n__g__" 혹은 "zhon" + " g__" 로 해석될 수도 있다. 그러나 실제로 중국어 병음에 "zho" 혹은 "zhon" 으로 이루어진 음절은 존재하지 않으므로(즉 당연히 병음색인에도 없음), "zhong..." 에 대하여 색인을 참조하여 "中" 으로 해석하게 된다. 만약 존재한다면 "zhong" 까지 입력되었을 때, 소정의 출력순서에 따라 후보 단어로써 출력할 수 있다. 도10-6을 기준으로 "zhongg... = 49***7733" 의 입력에 대하여, "zhonk.." = "zho" + "n__k__" 혹은 "zhon" + "k__" 로 해석될 수도 있도 있을 것이다. 이는 기 제시한 바 입력값을 풀코드와 단축입력값(심플코드)로 동시에 해석하는 병행입력방법과 유사한 맥락이다.

3> 이상의 예시에서 단축입력값(심플코드)으로 편의상 "음절기준 이니셜값 심플코드"를 사용한 사례를 제시하였으나, 단축입력값으로 "음절기준 이니셜코드" 적용되는 경우는 심플코드해석에 있어서 도11-3 ~ 도11-10에서 언급한 내용을 응용할 수 있다.

9> 26. 길게 누름을 이용한 문자입력방법 및 길게누름을 이용한 컨트롤처리 방법

10> 일반적으로 하나의 버튼 한번 누름으로 하나의 대상(예. 알파벳, 숫자, 영어를 모국어로 하지 않는 언어에서 영어알파벳, 기호(특수문자), 기능, 각종 용도의 컨트롤 등을 포괄하여 "알파벳 등" 이라 함)을 표현할 수 있다. 여기서 특정 버튼을 한번 누르되 "길게 누름"을 이용하여 단순히 한번 누름으로써 표현하는 대상과 다른 대상을 표현할 수 있다. 이하에서 특정 버튼을 길게 누르는 것을 "길게 누름" 혹은 "장타"라고 부르기로 한다. 이하에서 "길게 누름"이라고 표현하지 않는 것은 보통의 버튼 눌러짐을 의미하며, 특별히 길게 누름과 대비하여 설명할 때에는 "단타" 라고 한다. 이하에서 영어의 사례에서 일반적인 내용을 설명하고, 각 언어별로 응용 사례를 들어 설명하나, 어느 한 언어에서 설명한 내용이 타 언어에도 동일하게 적용될 수 있는 것은 자명하다.

361> 26.1 영어 (및 공통)

362> 26.1.1 길게 누름을 이용한 알파벳 입력

63> 현재 특정 숫자버튼을 길게 누름으로써, 해당 숫자를 입력하는 것이 사용되고 있다. 그러나 반드시 숫자만 입력가능한 것은 아니며, 예를 들어 도1-1에서 각 숫자버튼을 한번 누르는 것이 해당 버튼에 배정된 알파벳 중 첫번째 알파벳을 입력(예. [2]번버튼 1번 누름으로 "A"를 입력)하는 것이라고 가정할 때, 길게 누름에 의하여 두번째 알파벳(예. "B")를 입력하는 것으로 할 수 있다. 단 길게 누름은 입력의 자연스러운 흐름을 끊는 요인으로 자주 입력하는 알파벳의 입력에 응용하는 것은 바람직하지 않다. 따라서 길게 누름(장타)에 의한 입력은 자주 사용되지 않는 대상을 입력하는데 적용하는 것이 바람직하다. 또한 각 사용자의 숙련도에 따라 가능한한 길게 누름으로 인식되는 시간을 사용자로 하여금 설정할 수 있도록 하는 것이 바람직할 것이다.

64> 본 발명에서는 길게 누름을 적절하게 이용하되, 이를 통하여 특정 입력방법의 입력규칙을 단순화하고, 표현범위를 넓히고, 더 나아가 모호성을 제거할 수 있음을 보인다. 여기서 편의상 [2]버튼의 길게 누름을 "2" 다음에 물결표(~)를 붙여 "2~" 로 표기한다. 언급하였듯이, 기존의 반복선택방법 적용시, 도1-1에서 [2]버튼이 3번 눌러지면, "222" 가 "C" 인지, "AAA" 인지, "AB" 인지 "BA" 인지 알 수 없는 모호성이 발생한다.

65> 도1-1에서 "A" 는 [2]버튼 길게 누름 또는 해당 버튼 한번 누름, "B" 는 "A" 입력후 해당 버튼 한번 누름, "C" 는 "B" 입력후 해당버튼 한번 누름(혹은 "A" 입력후 2번누름도 같다)과 같이 정의하는 예를 든다. 같은 버튼에 있는 알파벳이 연속으로 입력되는 경우는 2번째 이후의 알파벳 입력에 있어서, 길게 누름으로 시작하면 역시 모호성 없이 입력할 수 있게 된다. 예를 들어 "DACB" 입력시 "3 2 2~22 2~2" 로 모호성 없이 입력할 수 있으며, 같은 버튼에 있는 "ACB" 가 연속하여 나오지만, "C" , "B" 입력시 길게 누름이 매개됨으로써 모호성없이 식별이 가능한 것이다. 이 사례는 기존의 반복선택방법과 거의 유사한 느낌으로 입

력할 수 있으며, 같은 버튼에 배정된 알파벳이 연속하여 입력되는 경우만, 길게누름을 이용하여 모호성을 제거할 수 있는 효과가 있다.

<36> 위의 사례는 특정 버튼에 배정(명시적 배정 및 암시적 배정 모두 포함)된 알파벳 중 임의의 알파벳(예. "A")을 대표알파벳으로 두고, 나머지 알파벳을 후속알파벳(예. "B" , "C")으로 두어 후속알파벳은 대표알파벳과 후속알파벳이 배정된 버튼을 소정횟수 누름에 의하여 입력하도록 하는 것으로 볼 수 있다. 즉, "A (대표알파벳) ? B(2nd 후속알파벳) ? C(3rd 후속알파벳)" 의 관계를 두어, "A = 2~" 으로, "B = A + 2 = "2~2" 로, "C = B + 2 = A+2 + 2 = 2~22" 로 하는 것이다. 또한 "A" 를 길게 누름이 아닌 보통의 누름(즉, 단타)로도 입력할 수 있다고 한 것은, 같은 버튼에 배정된 알파벳이 연속하여 입력되는 경우가 아니면, 대표알파벳의 입력에 있어서 기존의 반복선택방법에서와 같이 보통의 한번 누름(즉, 단타)으로 입력하는 것도 가능하기 때문이다. 이하에서 이를 편의상 "대표알파벳 길게 누름에 의한 반복선택방법" 이라 부른다.

<67> 26.1.2 길게 누름을 이용한 컨트롤처리방법

<68> 다음으로 컨트롤의 선택에 응용하는 사례를 보인다. 이하의 사례에서 컨트롤은 후입력되는 것을 가정하여 설명한다. 앞의 예에서, [2]버튼 단타, 장타 모두 "A" 를 표현하도록 한 것은 단지 기존의 반복선택방법과 유사한 느낌으로 입력할

수 있는 사례를 들기 위한 것이다. 그러나 특정 버튼 한번 누름에 의하여 특정 대상을 표현할 수 있고, 또 동일한 버튼의 길게 누름에 의하여 다른 대상을 표현할 수 있다. 예를 들어, 특정 버튼(예. [1]버튼)의 한번누름이 특정 대상(예. 임의의 "알파벳x" ? 여기서 "x" 는 실제 "x" 가 아닌 임의의 알파벳을 의미하는 것임)을 표현할 수 있고, 길게 누름이 다른 특정 대상(예. 임의의 "알파벳y")를 표현할 수 있다. 즉, "알파벳x = 1" , "알파벳y = 1~" 과 같이 되는 것이다.

69> 이는 컨트롤의 선택에서도 동일하게 적용될 수 있다. 예를 들어 임의의 버튼(예. 컨트롤버튼으로써의 [*] 버튼) 한번 누름이 특정 대상(예. 임의의 "컨트롤a1")를 표현할 수 있고, 동일한 버튼의 길게 누름이 다른 특정 대상(예. 임의의 "컨트롤b1")를 표현할 수 있다. 컨트롤버튼의 반복누름으로 "a2컨트롤" , "a3컨트롤" , ... 등을 선택할 수 있다고 하였다. 여기서 중요한 것은 컨트롤버튼의 길게 누름에 의하여 "b1컨트롤" 을 선택한 후, 길게 누름이 아닌 반복누름으로 "b2컨트롤" , "b3컨트롤" , "b4컨트롤" , ... 등을 선택될 수 있는 것이다. 역시 모호성은 발생하지 않는다. 이는 앞의 예시에서 대표알파벳 "A" 만을 길게 누름으로 입력하고, 나머지 후속알파벳 선택에 있어서, 단타를 조합하도록 한 사례를 참고할 수 있다.

370> 편의상 특정 버튼(예. [1]버튼)과 "b1컨트롤" 의 조합에 의하여 입력되는 대상을 "B1 " 과 같이 표시하면, 다음과 같이 된다.

671> "A1 = x + {a1컨트롤} = 1*" , "A2 = x + {a2컨트롤} = 1**" , "A3 = x + {a3컨트롤} = 1***" , . . . ,

72> "B1 = x + {b1컨트롤} = 1*~", "B2 = x + {b2컨트롤} = 1*~*", "B3 = x + {b3컨트롤} = 1*~**", . . .

73> 물론 b1, b2, b3 컨트롤의 선택에 각각 장타1회, 장타2회, 장타3회를 적용하는 것도 가능하다. 그러나, 길게 누름을 자주 사용하도록 하는 것이 좋지 않음은 너무나 당연한 사실이다. B1, B2, B3, ... 의 사례와 같이 컨트롤버튼 1회 길게 누름 이후 보통의 누름에 의하여 입력하는 것을 편의상 "길게 누름후 반복누름" 이라하고, 이러한 컨트롤처리방법을 "길게 누름후 반복누름에 의한 컨트롤처리방법" 이라고 한다. A1, A2, ... 계열은 기 제시한 컨트롤처리방법을 의미하고, B1, B2, ... 계열은 "길게 누름후 반복누름에 의한 컨트롤처리방법" 을 의미한다. 위 내용을 그래프 형식으로 나타내면 도14-1과 같다. 이상의 내용은 대표알파벳 길게 누름에 의한 반복선택방법에서와 같이 최초의 컨트롤 선택에만 한번의 길게누름을 적용하여 확장한 사례이다. 더불어 도14-2에서와 같이 추가적으로 길게 누름을 매개하여 다양한 확장(예. C계열, D계열)이 가능하다.

374> 여기서 컨트롤버튼이 아닌 버튼(예에서 [1]버튼)의 길게 누름을 적용(예. "알파벳y")하면, 또 다른 대상을 표현할 수 있음은 물론이다. 더 나아가 길게 누름을 이용한 반복선택방법(예. "A = 2~", "B = 2~2", "C = 2~22")으로 키패드 각 버튼에 표시된 알파벳(도1-1에서 3개의 알파벳)을 모호성없이 식별할 수 있다고 하였으므로, 숫자 버튼 조작에 의하여 선택할 수 있는 3가지 알파벳 선택후 컨트롤 버튼의 반복누름, 혹은 길게 누름후 반복누름에 의하여 각기 다른 그룹의 변형알파벳을 입력할 수 있는 것이다. 즉 도1-1에서 "A" 입력후 컨트롤 버튼의 반복누름에 의하여 특정 그룹(예. 그룹1)의 알파벳(기호, 숫자, 모국어, 영어 알파벳 등을 포괄하는 의미)을 입력하도록 할 수 있고, "길게 누름 후 반복누름" 에 의하여 또 다른

특정 그룹(예. 그룹2)의 알파벳을 입력하게 할 수 있다. 마찬가지로 "B" 입력과 컨트롤버튼의 반복누름 조합에 의하여 그룹3의 알파벳을 입력할 수 있으며, 길게 누름 후 반복누름에 의하여 또 다른 그룹인 그룹4의 알파벳을 입력할 수 있는 것이다. 도1-1 키패드 버튼상 표시된 3개 알파벳에 대하여 뿐만 아니라, 임의의 규칙(예. "2~222")을 정하고 이에 대하여 컨트롤버튼의 반복누름과 길게 누름후 반복누름 등에 의하여 임의의 그룹의 알파벳을 입력하도록 하는 것도 가능하다.

75> 길게 누름후 반복누름에 의한 컨트롤처리방법은 한국어의 사례(도4-5 ~ 도4-8) 및 일본어의 사례(도2-1, 2-2) 등등 에서와 같이 컨트롤버튼이 기본적인 모국어의 입력에 소요되는 경우, 문장 중에서 간헐적으로 사용되는 숫자의 입력 및 영어 알파벳(영어가 모국어가 아닌 언어에서)의 입력 등에 매우 유용할 것이다. 이하에서 각 언어별로 응용 사례를 간단히 언급하며, 언급하지 않은 다른 언어에도 적용될 수 있다.

76> 26.2 한국어

77> 도4-5 ~ 도4-8의 사례에서 변형알파벳(예. 격자음/경자음 및 탈락자음)을 기

본자음이 속한 버튼의 반복누름에 의하여 입력(예. "ㄱ=1", "ㄲ=11", "ㅋ=111" 또는 "ㄱ=1", "ㅋ=11", "ㄲ=111")하는 경우에도 유사하게 적용될 수 있다. 즉, 같은 버튼에 속한 알파벳이 연속으로 나오는 경우 2번째 이후에 나오는 알파벳 입력시, 길게 누름에 의하여 입력하는 것이다. 예를 들어 도4-5에서 [1]버튼의 연속누름을 "ㅋ" 으로 정의하면, "ㄱ" 이 연속하여 나오는 경우 "ㄱ+ㄱ" 이 아닌 "ㅋ" 으로 인식될 수 있는데, 이때에도 두번째 "ㄱ" 입력시 길게 누름으로 입력하면 "ㄱ+ㄱ" 으로 인식될 수 있다. 즉 "국가" 에서 두번째 음절의 "ㄱ" 입력시 길게 누름을 이용하는 것이다. "국카" 입력시에도 두번째 음절의 "ㅋ" 입력에 길게 누름을 이용하여 "ㅋ = 1~1" 로 입력하는 것이다. 이는 변형알파벳을 컨트롤처리방법에 의하여 입력하는 것과 선택적으로(2가지 중 한가지만, 혹은 2가지 모두 동시에) 적용될 수 있다.

78> 마찬가지로 모음과 함께 배정된 자음(예. 도4-5, 도4-6에서 [0]버튼에 배정된 "ㅎ")의 입력에 길게 누름을 응용할 수 있다. 즉 "ㅎ = 0~" 과 같이 되는 것이다. 출원인의 사견으로 선출원에서 기 제시한 바 도4-5를 기준으로 "ㅎ = 8**" 또는 "ㅎ = 0**" 또는 "ㅎ = 0 (선출원에서 기 제시한 특별한 경우만 가능)" 로 입력하는 것이 길게 누름을 이용하는 것보다 입력 효율면에서 좋을 것으로 짐작된다. 기 제시한 탈락자음(예. "ㅎ")의 입력방법 및 길게 누름에 의한 입력방법이 선택적으로 적용가능하다.

79> 다음으로 "길게 누름후 반복누름에 의한 컨트롤처리방법" 을 응용하는 사례를 제시한다. 출원인의 대표적인 한국어 입력시스템은 홈페이지 http://www.simplecode.net 에 시뮬레이터와 함께 소개(고어, 기호의 입력 및 병행입력기술까지 소개)되어 있으며, 더불어 숫자 및 영어 알파벳의 입력도 모드전환없이 가능함을 언급하고 있다.

0> 이를 본 업계의 관계자로부터 모드전환 없이 숫자 및 영어의 입력기술까지 제공하여 달라는 요구가 있어 어쩔 수 없이 본 출원을 실행하게 되었음을 밝힌다. (참고로 이러한 요청이 있었던 업체는 출원인의 한국어 입력기술을 채택하지 않는 것으로 되었다. 참으로 안타까운 일이다)

1> 입력에 있어서 일반적인 사용빈도는 "자국어 알파벳 ? 각종 기호(특수 문자) ? 숫자 ? 영어 알파벳" 의 순으로 볼 수 있다. 도4-5 등에서 보듯이 숫자 및 영어를 입력하는데 사용될 수 있는 컨트롤 버튼이 없는 상태이다. 여기서 숫자 버튼과 [*]버튼의 길게 누름(즉, "*~ = 숫자컨트롤")과 조합하여 숫자를 입력하는 것이 가능하다. 즉, "숫자 1 = 1*~" 로 하는 것이다. [*]버튼은 1차적으로 모음버튼이면서 동시에 컨트롤버튼으로 활용되고 있는데, [*]버튼 한번 누름은 모음 " ㅣ" 를 표하지만, [*]버튼 한번 길게 누름은 임의의 대상(예에서는 "숫자 컨트롤")을 표현할 수 있기 때문이다.

2> 영어 알파벳의 입력은 여러가지 방법이 가능하다. 예를 들어 "A = 2#~" , "B = 2#~#" , "C = 2#~#" 로 정의할 수 있다. "2#~#" 입력에 대하여 "A" 와 모음 " ㅣ" 로 인식되지 않고, "B" 로 인식될 수 있는 것은 것은 한국어에서 모음 " ㅣ" 로 시작되는 음절이 없는 성질(일종의 한국어 제한)을 이용하는 것이다. 만약 [#]버튼의 길게 누름 및 반복누름을 다른 용도로 사용한다면, 영어 알파벳을 숫자의 후속알파벳으로 간주하여 "A = 2*~*" 혹은 "A = 2*~# (교차 컨트롤처리방법)" 으로 "B = 2*~**" 혹은 "B = 2*~##" 등과 같이 다양하게 정의할 수 있다.

3> 길게 누름에 의한 컨트롤처리방법의 사례를 한가지 더 제시한다. 도4-5 ~ 도4-8에서, 모음 "ㅡ" 가 배정된 버튼의 반복누름에 의하여 "ㅜ" , "ㅠ" 를 입력하는 것(즉, "ㅡ = *" , "ㅜ = **" , "ㅠ = ***")로 정의하면, 알파벳 입력에 최대 3타(즉, 최대 가능 반복횟수

)의 [*]버튼 연속누름이 소요된다. 이 경우 격음컨트롤은 "최대가능 반복횟수 + 1" 회 누름(즉, [*]버튼 4회)에 의하여 선택될 수 있다. "즉, "ㅋ = 1****" 인 것이다. 이 경우 연속 4회 누르는 부담을 완화하기 위하여 "격음컨트롤 = *~" 로 하여 "ㅋ = ㄱ+{격음컨트롤} = 1*~" 로 정의할 수 있다. 이는 경자음 및 변형알파벳으로 간주하는 탈락자음에 대해서도 동일하게 적용할 수 있다.

34> 26.3 일본어

35> 도2-1, 도2-2의 일본어의 사례에서와 같이 후속알파벳이 다수인데, 반복선택방법에 의하여 후속알파벳을 입력시 유용할 수 있다. 즉 "あ = 1" 또는 "あ = 1~" 로, "い = 1~1", "う = 1~11", "え = 1~111", "お = 1~1111" 로 정의하는 것이다. 언급한 바와 같이 같은 버튼에 배정된 알파벳이 연속하여 나오는 경우, 같은 버튼에 배정된 알파벳 중 2번째 나오는 알파벳만, 길게 누름(즉, "1~")을 이용하여 입력하고, 그렇지 않은 경우는 단순히 한번 누름에 의하여 입력하여도 모호성없이 입력된다. 그러나, 도2-1, 도2-2의 일본어 사례에서 후속컨트롤처리방법을 적용함으로써 적은 입력타수로 모호성없이 입력할 수 있는 것을 이미 보였는데, 대표알파벳 길게 누름에 의한 반복선택방법과 함께 적용될 수 있다.

86> 대표알파벳 길게 누름에 의한 반복선택방법을 적용하지 않고, 컨트롤처리방법으로 후속알파벳을 입력하는 사례에서 길게누름을 이용하여 변형알파벳을 입력하는 사례를 보인다. 도2-2에서 컨트롤처리방법에 의하여 후속알파벳 입력시, 도2-3에서와 같이 대표알파벳(예. "は")의 변형알파벳(예. "ば" , "ぱ") 입력시, 입력타수가 다소 많고 부자연스러운 것을 알 수 있다. 이 경우, 해당버튼(도2-2에서 [6]버튼)의 길게 누름을 대표알파벳의 변형알파벳(

예. "ば")으로 정의할 수 있다. 나머지 변형알파벳 "ぱ = ば + 소정의 컨트롤버튼(예. [*] 버튼) = 6~*" 로 정의할 수 있다.

7> 여기서 알파벳이 배정된 버튼의 길게 누름을 이용하여 변형알파벳을 입력하지 않으면, 대표알파벳 길게 누름에 의한 반복선택방법과 컨트롤처리방법에 의한 후속알파벳 입력을 함께 적용할 수 있다. 변형알파벳(예. "ば", "ば")의 입력은 임의의 컨트롤버튼(예. [*]버튼)의 길게 누름과 대표알파벳 "は" 의 조합으로 "ば" 를 입력하도록 할 수 있다. 즉, "ば(임의의 변형알파벳) = は + *~ = 6*~" 으로, "ば(임의의 다른 변형알파벳) = は + *~* = 6*~*" 로 정의할 수 있다. 만약 "は" 의 변형알파벳이 다수인 경우도 "6*~**", "6*~***", ... 과 같이 길게 누름후 반복누름을 이용하여 모호성없이 입력하도록 할 수 있음은 물론이다. "*" 과 "*~" 이 각각 다른 대상을 표현할 수 있는 것처럼 "**" 과 "*~*" 이 각기 다른 대상(여기서는 임의의 컨트롤)을 표현할 수 있기 때문이다.

38> 여기서 중요한 것은, 길게 누름에 의하여 임의의 컨트롤을 선택할 수 있는 것뿐만 아니라, 한번 길게 누름 이후에 길게 누름에 이용된 버튼(예. 컨트롤버튼)의 반복누름에 의하여 모호성없이 다른 대상(예. 임의의 컨트롤)을 표현할 수 있는 것이다.

89> 마찬가지로 숫자 및 영어 알파벳의 입력에 있어서, 한국어에서 설명한 것을 응용할 수 있다. 예를 들어 "숫자 2 = 2#~" 으로 정의할 수 있으며, "A = 2#~#", "B = 2#~##", . . . 과 같이 정의할 수 있다.

90> 26.3.1 중국어

<1> 도10-1 ~ 도10-6 등의 키패드에서 모호성없이 (혹은 거의 모호성없이) 중국어 병음을 입력할 수 있음을 보였다. 그러나 "b__b__" 로 구성된 단어(밑줄 부분은 모음으로 구성된 병음음절)를 단축입력하기 위하여 "bb = 11" 을 입력시, 중국어 병음에서 "bb" 가 연속하여 나오지 않는 언어제한이 적용되면, "11 = p" 로 인식될 것이다. 이 경우는 언급한 바와 같이, "b = 1" 을 입력후 소정의 시간지연 혹은 특정 조작(예. [>]버튼 누름)에 의하여 의도적으로 입력값 "1" 을 "b" 로 확정 후 "b = 1" 을 입력하여야 한다. 역시 같은 버튼에 배정된 알파벳을 연속하여 입력할 경우 발생하는 모호성의 문제로 볼 수 있다. 이 경우도 대표알파벳 길게 누름에 의한 컨트롤처리방법을 적용하여 ([1]버튼의 대표알파벳 "b" 를 "b = 1~" 로) 해결할 수 있다. 예를 들어 "bb = 1~1~" 혹은 같은 버튼에 배정된 알파벳이 연속으로 나올 때 2번째 이후에 있는 알파벳 입력에 길게 누름을 적용하여 "bb = 11~" 로 할 수 있는 것이다.

<92> 다음으로 숫자와 기호를 입력하는 것을 보인다. 도10-6을 기준으로 설명한다. 중국어에서 영어의 입력은 영어 알파벳 입력후 이를 한자로 변환하지 않으면 되므로 굳이 언급하지 않는다. 도10-6을 기준으로 "숫자 1 = 1*~" 과 같이 정의할 수 있다. 도10-6에서 [*]버튼 한번 누름은 "a" 로 인식하게 되나, 길게 누름("*~")은 별개의 대상(예에서 "숫자 컨트롤")을 표현할 수 있기 때문이다. 도10-6에서, 각종의 기호 입력에는 [#]버튼의 길게 누름 및 반복누름을 이용한 컨트롤처리방법을 적용하는 것이 바람직하다. 예를 들어 [2]버튼에 dot(.), comma(,), colon(:), semi-colon(;) 등을 연관시키고, "dot(.) = 2#~" , "comma(,) = 2#~#" , "colon(:) = 2#~##" , "semi-colon(;) = 2#~###" 으로 정의할 수 있다.

"2#~#" 이 "dot(.) + i" 로 인식되지 않고 "comma(,)" 로 인식될 수 있는 것은 중국어 병음이 "i" 로 시작하지 않는 중국어 제한을 이용하기 때문이다. (한국어의 사례 참조) 마찬가

지로 "2#~##" 도 "dot(.)+i+i" 혹은 "comma(,)+i" 로 인식되지 않고 "colon(:)" 으로 인식될 수 있다. 만약, "dot(.)+i" 를 입력하고 싶으면, 언급한 바대로, "2#~" 까지 입력후 소정의 시간 경과로 "dot(.)" 확정 후 혹은 특정 조작(예. [>]버튼 누름)에 의하여 "dot(.)" 확정후 "# = i" 를 입력하면 된다.

<93> 도10-6에서 특정 그룹의 기호(들)의 입력에 [*]버튼을 사용하지 않은 이유는, 드물지만 "a", "e", "o" 등으로 시작하는 병음이 있으므로, "길게누름 후의 반복누름" 을 이용시, 모호성이 발생할 수 있기 때문이다. 숫자의 경우는 하나씩만 존재하므로 [*]버튼 길게 누름 한번만 조합하면 되기 때문이다. 도10-1 ~ 10-5의 경우, [*]버튼과 [#]버튼에 배정된 "i" 와 "u" 로 시작하는 병음이 없으므로, [*]버튼과 [#]버튼을 이용하여 길게누름 후 반복누름을 이용한 컨트롤처리방법으로 숫자, 기호, 등을 입력하는 것이 항상 가능하다.

<94> 이상의 내용은 도10-1 ~ 도10-6의 키패드 뿐만 아니라 다른 키패드에서도 유사하게 적용될 수 있음은 자명하다.

【발명의 효과】

<95> 본 발명에서 자모음분리키패드에서 (언어제한) 반복선택방법을 적용하여, 어떤 언어에서는 모호성없이 혹은 대부분의 언어에서 적은 모호성으로 입력하고자 하는 어구를 입력할 수 있음을 보였다.

6> 또한 언어제한 입력방법을 풀입력방법으로 하는 병행입력방법(즉, 언어제한 병행입력방법)을 적용하여 특정 어구의 입력초기에 입력값이 심프코드인지 아니면 풀코드인지를 시스템이 효율적으로 인식하여 처리할 수 있는 효율적인 시스템을 제시하였다.

【특허청구범위】

【청구항 1】

키패드에서 반복선택방법으로 알파벳을 입력함에 있어서, 연타지연시간(동일버튼 2타를 2번째 알파벳으로 1차적으로 인식하도록 하는 시간)과 이타지연시간(동일버튼 2타를 1번째 알파벳의 2번입력으로 1차적으로 인식하도록 하는 시간)을 각각 다르게 설정하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 2】

키패드에서 알파벳을 입력하는 방법에 있어서, 알파벳을 임의의 그룹으로 나누어 키패드 내의 임의의 버튼에 배정하고, 해당 그룹의 알파벳에 대표알파벳과 후속알파벳의 관계를 두어, "다음컨트롤"을 임의의 컨트롤버튼누름으로 선택할 수 있도록하며, 후속알파벳입력시 선행알파벳과 "다음컨트롤"을 조합하여 후속알파벳을 입력하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 3】

키패드에서 컨트롤처리방법을 통하여 알파벳을 입력함에 있어서, 대표알파벳과 결합되는 컨트롤이 의미를 가지지 않는 경우(즉 선택된 컨트롤과 대표알파벳의 조합이 의미있는 알파벳을 이루지 못하는 경우), 해당 컨트롤이 존재하지 않는 것으로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 4】

키패드에서 컨트롤처리방법을 통하여 알파벳을 입력함에 있어서, 각 버튼에 배정된 숫자와 영어알파벳을 후속알파벳으로 간주하여 컨트롤처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 5】

키패드를 통한 아랍어 알파벳 입력에 있어서, 키패드상 1로 시작하는 숫자(1, 10, 100 혹은 1, 10, 100, 1000)의 의미를 가지는 알파벳(이하 "1그룹알파벳"이라 약칭), 2단위의 숫자(2, 20, 200)의 의미를 가지는 알파벳(이하 "2그룹알파벳"이라 약칭), 이하 같은 식으로 각 단위의 숫자의 의미를 가지는 알파벳을 약 3개씩 9개의 그룹으로 그룹핑하고, 각각의 그룹을 [1] ~ [9]의 9개 버튼에 배정하고,

각 그룹에서 1 ~ 9의 숫자의 의미를 가지는 알파벳(이하 "1단위알파벳"이라 약칭), 10 ~ 90의 의미를 가지는 알파벳(이하 "10단위알파벳"이라 약칭) 및 100 ~ 900의 숫자(이하 "100단위알파벳"이라 약칭) 중 임의의 단위의 알파벳을 대표알파벳으로 정하고, 나머지 단위의 알파벳을 후속알파벳으로 간주하여 컨트롤처리하여 입력하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 6】

캐패드를 통하여 한국어를 입력함에 있어서, 10개의 기본자모를 각각 쌍으로 그룹핑하고 자음1타, 모음2타로 선택하고, 격음컨트롤을 [*], [#] 버튼 중 임의의 버튼에 두고, 경음컨트롤을 격음컨트롤이 배정되지 않은 버튼에 두어 경음과 격음을 컨트롤처리하여 입력하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 7】

자음과 모음을 쌍으로 그룹핑하고 키패드상의 각 버튼에 배정하고 각 버튼 1타로 키패드상의 자음을 선택하고 각 버튼 2타로 키패드상의 모음을 선택하는 방법에 있어서, 모호성이 발생하는 가장 많은 경우인 모자전이("자+모+자" 로 이루어진 음절에서 모음과 자음의 결합)와 자모전이("자+모"로 이루어진 음절에 있어서 자음과 모음의 결합)시의 빈도수가 전체적으로 최소화 되도록 자음과 모음을 그룹핑하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 8】

한국어에 있어서, 키패드 버튼상에 배치된 알파벳 혹은 컨트롤을 반복선택방법으로 선택하여 입력함에 있어서, 경음과 격음이 존재하지 않는 기본자음 중 하나를 제외한 나머지 9개의 기본자음을 [1] ~ [9] 의 숫자 버튼에 임의로 배정하고, 3개의 모음 알파벳요소(ㅡ, ., ㅣ)를 나머지 3개의 버튼([*], [0], [#])에 임의로 배정하고 각 배치된 기본자음과 모음 알파벳 요소를 1타에 선택할 수 있도록 배치하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 9】

제8항에 있어서, 경음과 격음을 컨트롤처리함에 있어서, "ㅡ" 혹은 "ㅣ" 가 배정된 버튼 중 임의의 버튼 2타로 격음컨트롤이 선택되고, 나머지 버튼 2타로 경음컨트롤이 선택되도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 10】

제9항에 있어서, [1] ~ [9]의 숫자버튼에 배정되지 않은 기본자음을 경음과 격음이 존재하지 않는 임의의 기본자음의 변형알파벳(격음 혹은 경음)으로 간주하여 컨트롤처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 11】

제10항에 있어서, [1] ~ [9] 버튼에 배정되지 않은 기본자음을 "."이 배정된 버튼에 추가로 배정하고 해당 버튼 3타로 선택할 수 있도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 12】

제11항에 있어서, 경음을 기본자음의 조합으로 입력하도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 13】

제12항에 있어서, 격음을 해당 기본자음이 속한 버튼 3타로 입력하도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 14】

키패드를 통한 알파벳 입력에 있어서, 풀코드 입력시 모호성이 발생할 경우, 단어단위로(즉 단어종료시점에) 클라이언트측 혹은 서버측에 구비된 색인을 검색하여 올바른 단어를 식별해 내는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 15】

제14항에 있어서, 1차적으로 클라이언트측의 색인을 검색하고, 2차적으로 서버측의 색인을 검색하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 16】

심플코드를 활용함에 있어서, 어구(단어 혹은 구절)에 포함된 자음과 연관하여 키패드상 대응되는 숫자를 심플코드로 정하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 17】

심플코드를 활용함에 있어서, 어구의 음절을 이루는 첫자음 혹은 모음과 연관하여 키패드상 대응되는 숫자를 심플코드로 정하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 18】

심플코드를 활용함에 있어서, 어구를 이루는 단어의 첫 알파벳과 연관하여 키패드상 대응되는 숫자를 심플코드로 정하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 19】

제16항 ~ 제18항에 있어서, 어구내에서 심플코드와 연관된 알파벳을 부각하여 표시하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 20】

제19항에 있어서, 영어의 경우 대문자를 사용하여 어구내에서 심플코드와 연관된 알파벳을 부각하여 표시하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 21】

제16항 ~ 제19항에 있어서, 심플코드 입력시 입력값에 대응되는 알파벳을 클라이언트측 혹은 서버측에서 검색하여 사용자에게 제공함으로써 어구 입력을 가능하게 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 22】

제21항에 있어서, 심플코드 및 심플코드에 대응되는 어구를 트리구조로 그룹핑하고 심플코드의 검색범위를 제한하였을 경우, 입력값에 대하여 제한된 범위의 심플코드 및 대응어구만을 검색하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 23】

제22항에 있어서, 기본입력모드를 풀입력모드로 설정하였을 경우, 입력값을 1차적으로 설정될 풀입력방법에 의하여 해석하고, 약속된 풀코드 생성규칙을 위배하는 순간, 입력값을 심플코드로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 24】

제23항에 있어서, 서버측으로부터 심플코드 및 심플코드 대응어구의 그룹을 클라이언트측으로 다운로드 받는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 25】

제24항에 있어서, 심플코드의 해석을 담당하는 중계서버를 두는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 26】

제25항에 있어서, 사용자로부터 입력된 심플코드를 중계서버가 해석하여 심플코드에 대응되는 어구를 최종애플리케이션을 탑재한 서버(3차서버)측에 전달해 주는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 27】

키패드를 통하여 각종 기호를 입력함에 있어서, 기호를 임의의 그룹으로 나누고, 각 그룹을 키패드내의 임의의 버튼에 특정 기호그룹의 의미를 부여하고(특정 기호그룹과 연관시키고), 임의의 버튼에 다수의 기호컨트롤(기호1, 기호2, ...)을 두고 기호컨트롤을 반복선택방법에 의하여 선택하며, 특정 기호그룹의 의미를 부여한 버튼과 기호컨트롤들을 조합하여 특정 기호그룹의 기호들을 입력하게 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 28】

상하좌우 이동버튼이 구비된 단말기를 통하여 알파벳을 입력함에 있어서, 알파벳입력모드에서 빈번히 사용되지 않는 상하좌 이동버튼을 각종 컨트롤버튼으로 활용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 29】

제28항에 있어서, 계산기모드에서 상하좌우 이동버튼을 가감승제버튼으로 활용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 30】

각종 컨트롤처리방법을 통하여 키패드에서 알파벳을 입력함에 있어서, 컨트롤버튼의 기능을 요약하여 아이콘화하여 스크린(액정)의 일부분에 표시하여 주는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 31】

제30항에 있어서, 숫자버튼에 연관된 기호그룹에서 일부의 기호와 숫자를 아이콘화하여 스크린(액정)의 일부에 표시하여 주는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 32】

심플코드를 활용함에 있어서, 동일한 심플코드에 대응되는 어구가 다수인 경우, 사용자가 기 설정된 우선순위와 다르게 최종선택하는 경우가 일정기준 이상 발생할 경우, 기 설정된 우선순위를 재조정하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 33】

표준 키보드상 구비된 숫자키패드의 숫자버튼배치를 전화기상 구비된 숫자버튼배치와 동일하게 통일하는 것을 특징으로 하는 키보드

【청구항 34】

키패드를 통하여 알파벳을 입력하는 방법에 있어서, 단축입력방법과 풀입력방법을 병행하는 입력방법을 병행하여 적용하되, 단어의 입력중간에라도 입력값이 기본입력모드로 설정된 입력방법의 입력값이 아닌 것으로 판단되는 순간, 시스템이 입력값을 기본입력방법이 아닌 타 입력방법의 입력값으로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 35】

제34항에 있어서, 기본입력모드를 단축입력모드로 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법(단축우선 병행입력방법)

【청구항 36】

제34항에 있어서, 기본입력모드를 풀입력모드로 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법(풀우선 병행입력방법)

【청구항 37】

제36항에 있어서, 입력값이 특정 언어의 언어제한(즉, 단어생성규칙 또는 알파벳결합규칙)을 위배하여 특정언어의 유효한 음절을 이루지 못하는 것을 시스템이 인식하는 순간, 시스템이 입력값을 심플코드로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법 (언어제한 병행입력방법)

【청구항 38】

알파벳을 입력하는 방법에 있어서(키패드 혹은 PC 등 모든 정보통신장치), 입력값이 특정 언어의 언어제한(즉, 단어생성규칙 또는 알파벳결합규칙)을 위배하여 특정언어의 유효한 음절을 이루지 못하는 것을 시스템이 인식하는 순간, 사용자가 단축입력을 하는 것으로 시스템이 인식하고 처리하는 것을 특징으로 하는 알파벳 입력방법

【청구항 39】

제37항에 있어서, 입력값을 심플코드로 간주하여 처리하되, 심플코드 색인에 입력값과 일치하는 심플코드가 더 이상 존재하지 않는 것을 시스템이 확인하는 순간 입력값을 다시 풀코드로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 40】

제37항에 있어서, 입력값(숫자값)을 심플코드로 간주하여 처리하되, 심플코드와 심플코드 대응어구가 저장된 심플코드 색인을 참조하여 일치하는 어구를 검색하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 41】

제38항 또는 제39항 중 어느 한 항에 있어서, 심플코드 대응문자를 심플코드 색인에 미리 등록하여 두고, 시스템이 이 심플코드 대응문자와 입력값이 생성하는 특정언어의 유효한 음절을 이루지 않는 단어(즉, 심플코드 대응문자)를 비교하여 타겟어구를 검색해 내는 것을 특징으로 하는 알파벳 입력방법

【청구항 42】

제38항 또는 제39항 중 어느 한 항에 있어서, 검색대상이 되는 심플코드 색인에 특정형태의 심플코드(예. 음절기준 이니셜코드)만이 저장되어 있을 경우, 입력값이 생성하는 특정언어의 유효한 음절을 이루지 않는 단어(즉, 심플코드 대응문자)와 색인에 저장된 타겟어구 후보어구과 비교하여 각 음절의 첫 자음이 일치하는 어구를 검색해 내는 것을 특징으로 하는 알파벳 입력방법

【청구항 43】

키패드를 통하여 알파벳을 입력하는 방법에 있어서, 반복선택방법을 적용하되, 특정언어의 언어제한(즉, 자음과 모음이 결합되는 단어생성규칙 또는 알파벳결합규칙)을 위반하는지를 시스템이 검사하여, 결합이 불가능한 알파벳조합을 제외함으로써, 반복선택방법적용시의 모호성을 감소시키는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 44】

제43항에 있어서, 자음이 배정되는 버튼(자음버튼)과 모음이 배정되는 버튼(모음버튼)을 분리하여 자모음분리키패드를 구성하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 45】

제44항에 있어서, 로마알파벳의 자모음 분리키패드를 구성하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 46】

제45항 또는 제46항 중 어느 한 항에 있어서, 3*4 키패드에서 9개의 버튼에 자음을 배정하고, 나머지 3개의 버튼에 모음을 배정하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 47】

제37항, 제38항, 제43항 ~ 제46항 중 어느 한 항에 있어서, 영어에서 "sp~" 혹은 "st~" 로 시작하는 경우를 제외하고는 "단어시작" 시부터 연속하여 3개의 자음이 나올 수 없는 언어제한(단어생성규칙)을 이용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 48】

제37항, 제38항, 제43항 ~ 제46항 중 어느 한 항에 있어서, 인도네시아어에서 "sp~" 혹은 "st~" 로 시작하는 경우를 제외하고는 "단어시작" 시부터 연속하여 3개의 자음이 나올 수 없는 언어제한(단어생성규칙)을 이용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 49】

제37항, 제38항, 제43항 ~ 제46항 중 어느 한 항에 있어서, 로마알파벳을 사용하는 언어에 있어서, 영어의 경우와 유사하게, "sp~" 혹은 "st~" 로 시작하는 경우를 제외하고는

"단어시작" 시부터 연속하여 3개의 자음이 나올 수 없는 언어제한(단어생성규칙)을 이용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 50】

제37항, 제38항, 제43항 ~ 제46항 중 어느 한 항에 있어서, 중국어 및 일본어를 로마알파벳으로 표기함에 있어서, 로마알파벳 자음이 연속하여 3개이상 나오지 않는 언어제한(단어생성규칙)을 이용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 51】

제37항, 제38항, 제43항 ~ 제50항 중 어느 한 항에 있어서, "일시적 언어제한해제 지연시간" 을 설정할 수 있도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 52】

키패드를 통하여 알파벳을 입력하는 방법에 있어서, 단축입력방법과 풀입력방법을 병행하는 입력방법을 병행하여 적용하되, 검색대상이 되는 심플코드 색인에 저장된 심플코드의 길이가 일정 개수 이하일 경우, 입력값의 길이로 입력값이 심플코드인지 풀코드인지를 판단하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 53】

키패드를 통하여 알파벳을 입력하는 방법에 있어서, 컨트롤처리방법을 적용하되, 알파벳, 기호, 혹은 기능의 입력에 기 사용된 컨트롤버튼을 다시 다른 용도의 컨트롤버튼으로 사용하는 다차원 교차컨트롤처리방법을 적용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 54】

제53항에 있어서, 알파벳 입력에 N차원 컨트롤처리방법이 적용되고 있을 때, N+1회 컨트롤버튼을 번갈아 조합함으로써, 변형알파벳 및 각종 기능을 입력하도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 55】

키패드를 통하여 한국어를 입력하는 방법에 있어서, 10자음 3모음법을 적용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 56】

키패드를 통하여 한국어를 입력하는 방법에 있어서, 10자음 4모음법을 적용하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 57】

제55항 ~ 제56항 중 어느 한 항에 있어서, 9버튼탈락자음 혹은 8버튼탈락자음을 모음요소와 함께 배정하고, 반복선택방법에 의하여 선택하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 58】

제55항 ~ 제56항 중 어느 한 항에 있어서, 9버튼탈락자음 혹은 8버튼탈락자음을 변형알파벳으로 간주하여 기본자음과 변형알파벳컨트롤(예. 격음컨트롤)과 조합하여 컨트롤처리방법에 의하여 입력하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 59】

제55항 ~ 제56항 중 어느 한 항에 있어서, 9버튼탈락자음 혹은 8버튼탈락자음을 모음요소와 함께 배정하고, 배정된 버튼과 모음요소 "ㅡ" 와 함께 배정된 컨트롤과 조합하여 선택하도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 60】

제55항 ~ 제56항 중 어느 한 항에 있어서, 경음과 격음을 반복선택방법에 의하여 소정의 순서에 따라 선택하도록 하는 것을 특징으로 키패드에서의 알파벳 입력방법

【청구항 61】

단축입력 및 심플코드의 활용을 위한 심플코드를 지정함에 있어서, "전체(알파벳)연관 심플코드", "음절기준 이니셜코드", "자음연관 심플코드", "첫모음+자음연관 심플코드", 혹은 "단어기준 이니셜코드" 를 선택적으로 이용하여 심플코드로 지정하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 62】

단축입력 및 심플코드의 활용을 위한 심플코드를 지정함에 있어서, 특정 어구에 대한 심플코드값을 사용자가 임의로 지정할 수 있도록 하는 것을 특징으로 하는 키패드에서의 심플코드 활용방법

【청구항 63】

제37항, 제38항, 제43항 ~ 제46항 중 어느 한 항에 있어서, 모음의 수가 모음을 배정할 버튼의 수보다 적은 경우, 복수개의 모음을 그룹으로 그룹핑하고 모음버튼에 배정하고 배치함

에 있어서, 동일한 모음이 연속하여 나오는 빈도가 적은 모음을 배정된 버튼 1타로 선택하도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 64】

제37항, 제38항, 제43항 ~ 제46항 중 어느 한 항에 있어서, 중국어 한어병음을 입력하기 위하여 a, e, i, o, u,..u 6개의 모음을 그룹핑하고 배정함에 있어서, 중국어 한어병음의 로마알파벳 모음결합규칙을 이용하여 "a" 와 "i" , 그리고 "e" 와 "i" 그리고 o 와 u 가 같은 그룹으로 그룹핑되지 않도록 그룹핑(예. a, e / i, o / u,..u 로 그룹핑)함으로써, 반복선택방법으로 모호성 없이 입력할 수 있도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 65】

제37항, 제38항, 제43항 ~ 제46항 중 어느 한 항에 있어서, 중국어 한어병음을 입력하기 위하여 a, e, i, o, u, 5개의 모음을 그룹핑하고 배정함에 있어서, 중국어 한어병음의 로마알파벳 모음결합규칙을 이용하여 i / u 를 각기 그룹핑하여 다른 버튼에 분리하고, 나머지 3개의 모음(a, e, o)을 또 다른 버튼에 배정함으로써, 반복선택방법으로 모호성없이 선택할 수 있도록 하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 66】

제65항에 있어서, "i, a, e" 위에 성조부호 등이 붙는 변형알파벳을 입력하기 위하여 "u" 가 배정된 버튼을 컨트롤버튼처럼 이용하고, "u, o" 위에 성조부호 등이 붙는 변형알파벳을 입력하기 위하여 "i" 가 배정된 버튼을 컨트롤버튼처럼 이용하여, 변형알파벳을 입력하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 67】

제65항에 있어서, "..u" 를 "u" 가 배정된 버튼 2타로 입력하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 68】

표준 키보드상 구비된 숫자키패드에 알파벳을 부기하여 심플코드의 활용에 이용할 수 있도로 하는 것을 특징으로 하는 키보드

【청구항 69】

키패드를 통하여 알파벳을 입력하는 방법에 있어서, 한자변환버튼을 반복적으로 눌러 후속한자를 입력하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 70】

키패드를 통하여 알파벳을 입력하는 방법에 있어서, "히라가나 ? 가타가나 - 대응한자1 - 대응한자2, …" 의 순서로(혹은 가타가나 입력모드에서는 "가타가나 - 히라가나 - 대응한자1 - 대응한자2, …" 의 순서로) 버튼 누름의 반복선택에 의하여 선택하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 71】

키패드를 통하여 알파벳을 입력함에 있어서, 심플코드를 구축함에 있어서, 사용자가 입력한 어구(단어 혹은 구절)로부터 자동으로 심플코드 색인을 구축하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 72】

키패드에서 알파벳을 입력하는 방법에 있어서, 시스템에 특정 풀입력방법에 따른 풀코드를 색인에 저장하고 있으면서, 입력값과 색인의 풀코드를 비교하여 타겟어구를 해석해내는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 73】

제72항에 있어서, 입력값의 입력에 따라 특정 후보 알파벳조합이 더 이상 색인에 존재하지 않는 것을 확인하는 순간, 특정 가능알파벳조합(즉, 후보알파벳조합)을 이후의 입력값에 대하여 후보알파벳조합에서 제외하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 74】

키패드에서 알파벳을 입력하는 방법에 있어서, 입력값을 동시에 약속된 풀코드 혹은 심플코드(즉, 단축코드, 단축입력값)으로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 75】

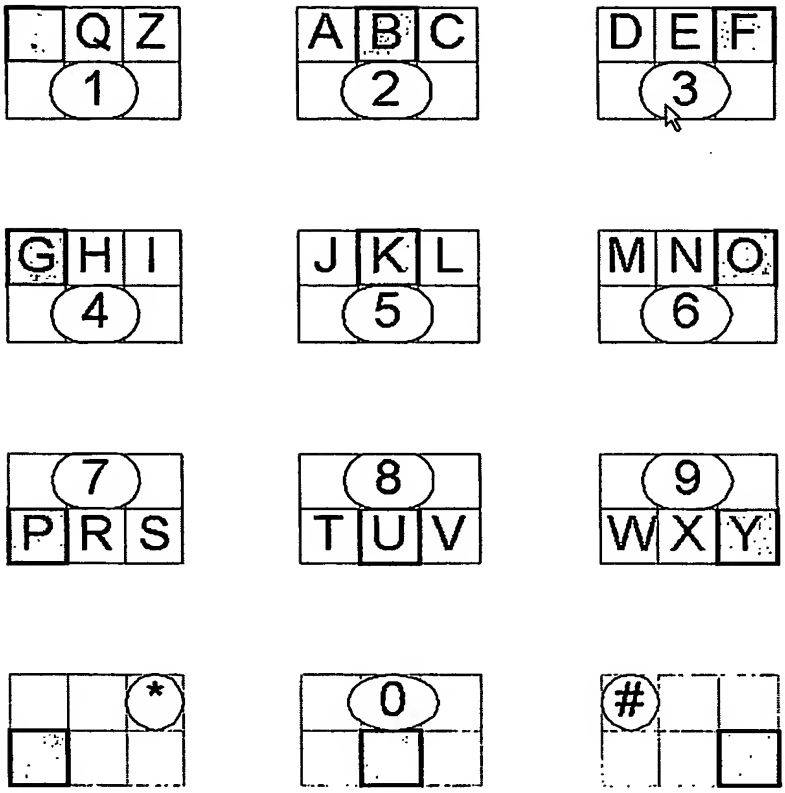
키패드에서 알파벳을 입력하는 방법에 있어서, "컨트롤버튼을 포함하지 않은 연속된 입력값" 에 대하여 입력값을 풀코드로 간주하여 처리하되, 동시에 상기 입력값이 생성하는 문자열이 대표알파벳만으로 이루어진 모든 단어의 색인(혹은 이러한 색인을 포함하고 있는 색인)에 존재하는지 확인하여 대표알파벳만으로 이루어진 단어가 색인에 더 이상 존재하지 않으면 상기 입력값을 단축코드로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【청구항 76】

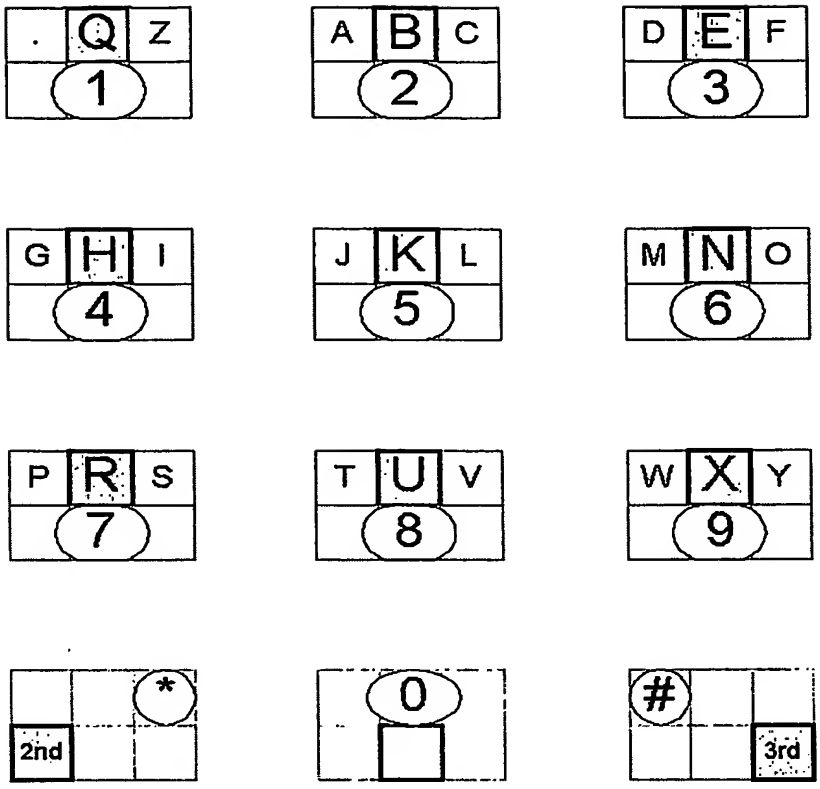
제75항에 있어서, 상기 입력값이 생성하는 문자열이 대표알파벳만으로 이루어진 모든 단어의 색인(혹은 이러한 색인을 포함하고 있는 색인)에 더 이상 존재하지 않는 것이 확인되는 순간 상기 입력값을 단축코드로 간주하여 처리하는 것을 특징으로 하는 키패드에서의 알파벳 입력방법

【도면】

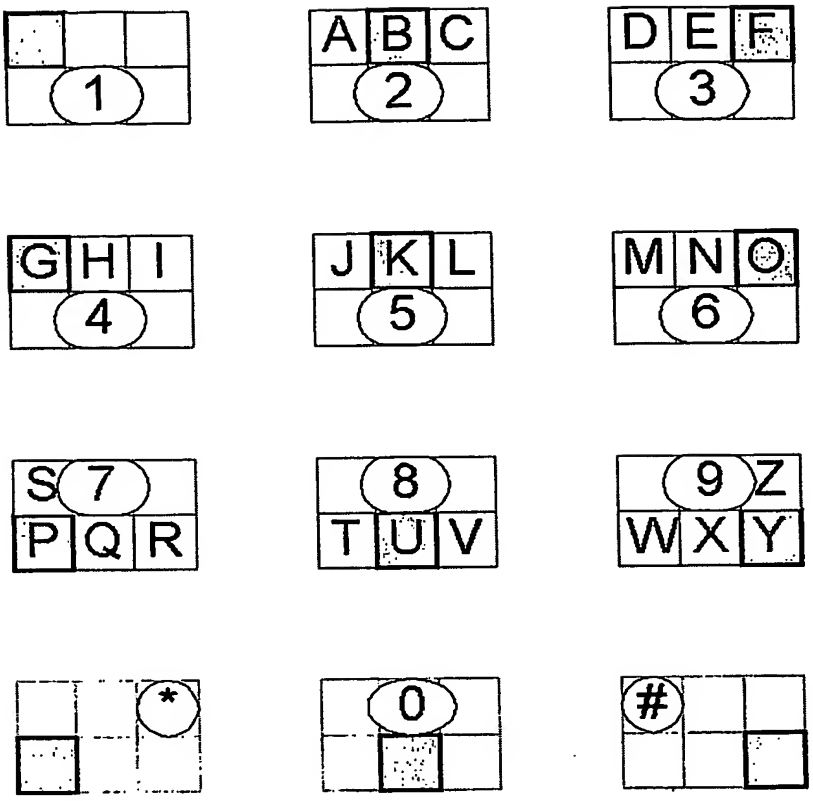
【도 1】
도 1-1



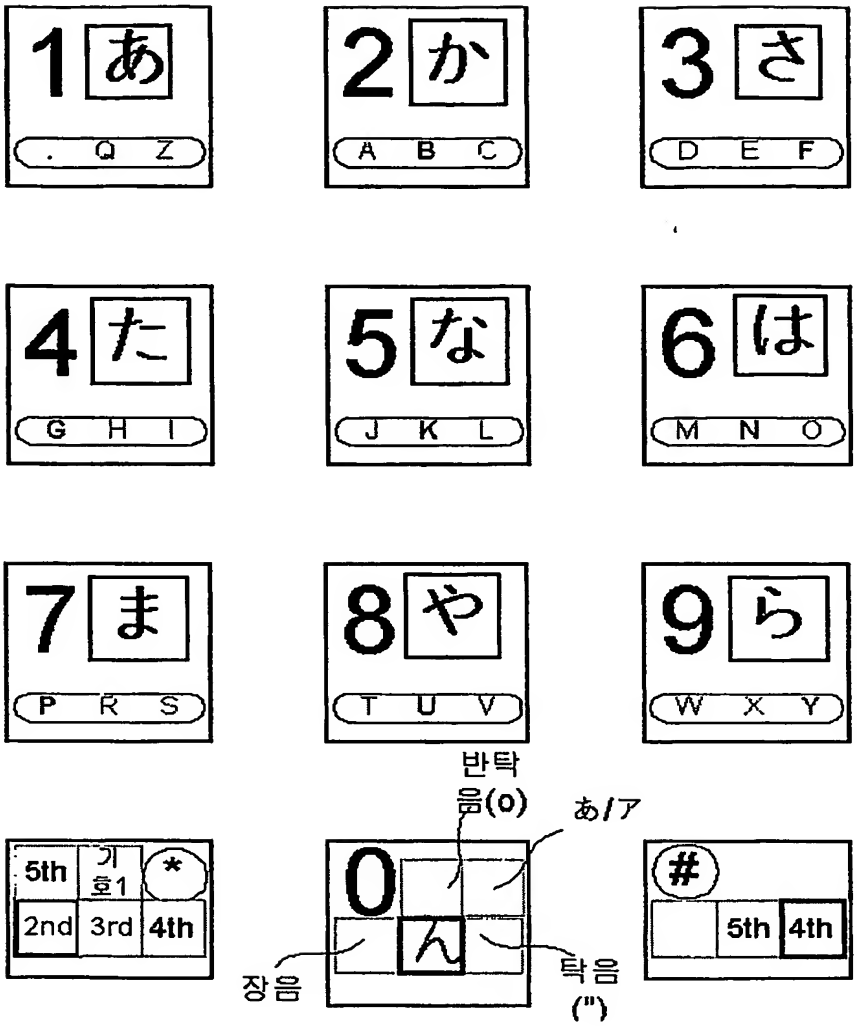
【도 2】
도 1-2



【도 3】
도 1-3

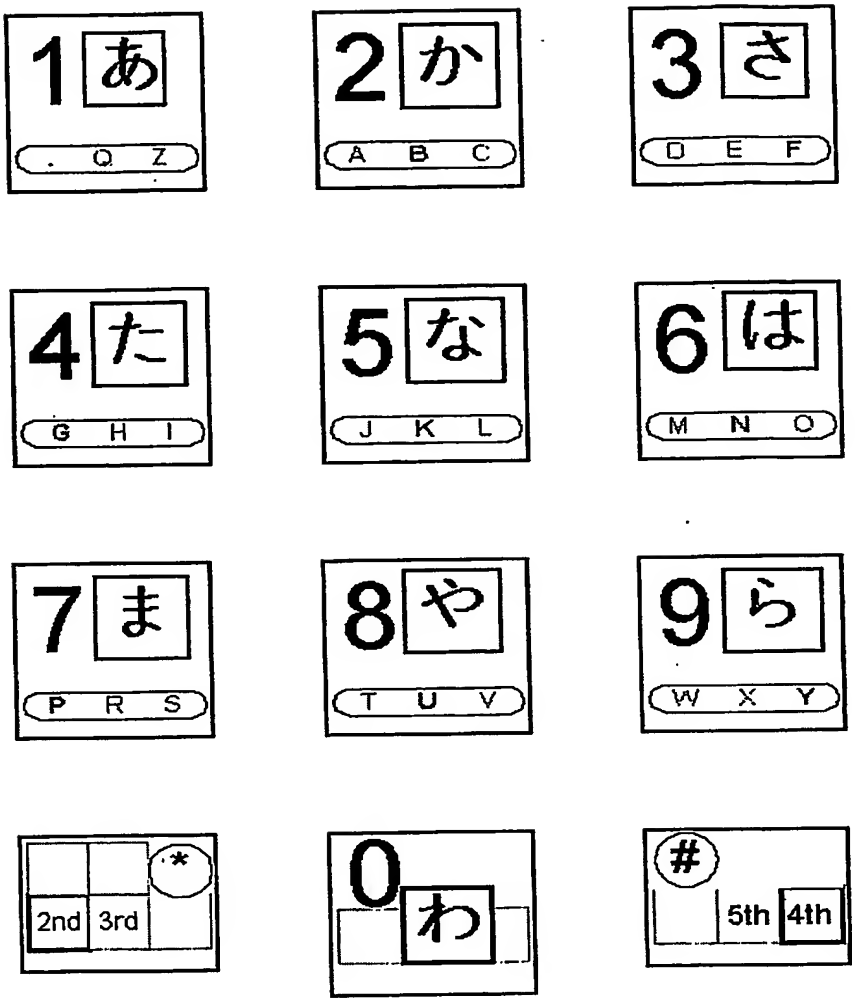


【도 4】
도 2-1


1 あ
. Q Z
2 か
A B C
3 さ
D E F
4 た
G H I
5 な
J K L
6 は
M N O
7 ま
P R S
8 や
T U V
9 ら
W X Y
반탁음(0)
あ/ア
5th
기호1
*
2nd
3rd
4th
0
ん
장음
탁음(")
#
5th
4th

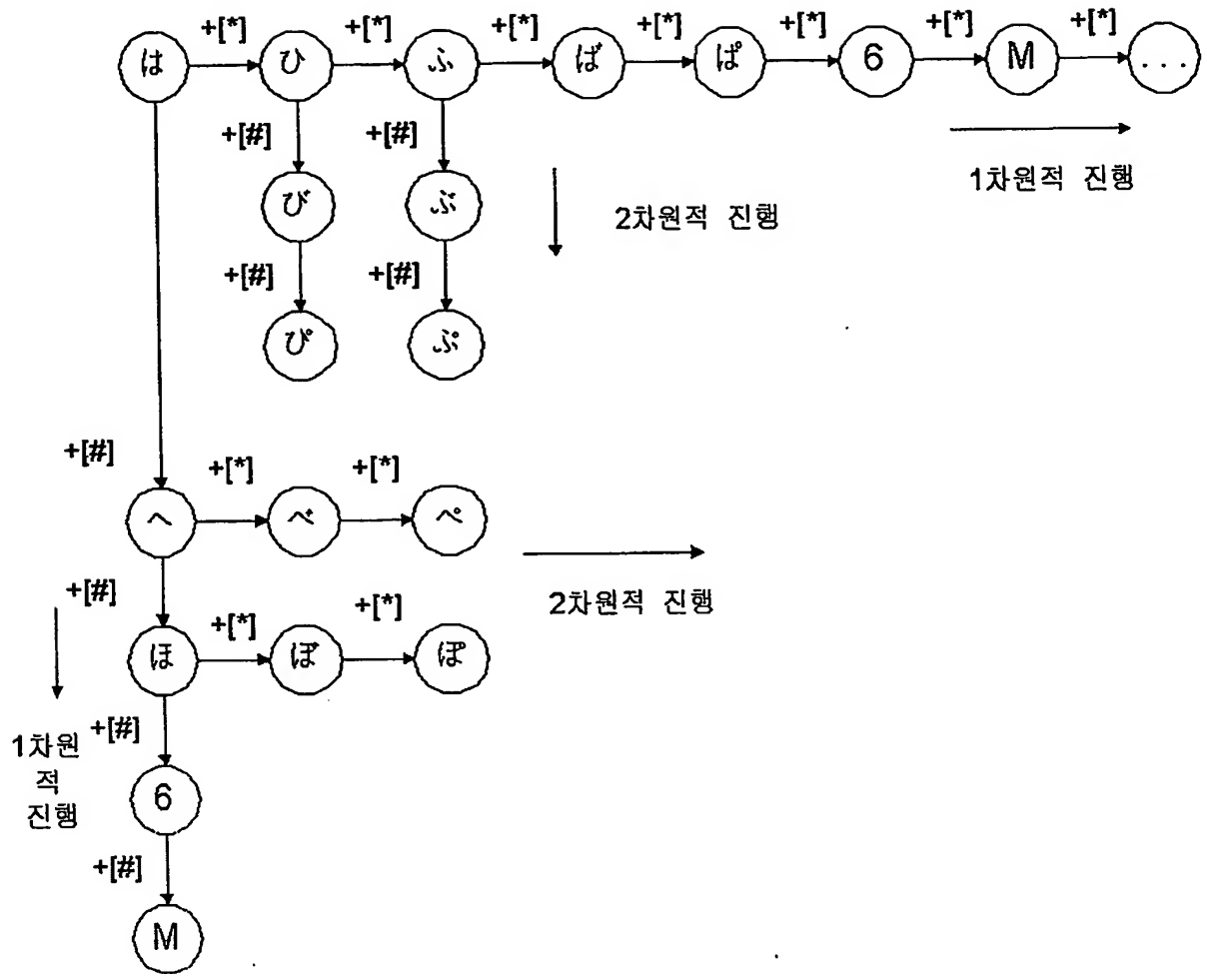
【도 5】

도 2-2

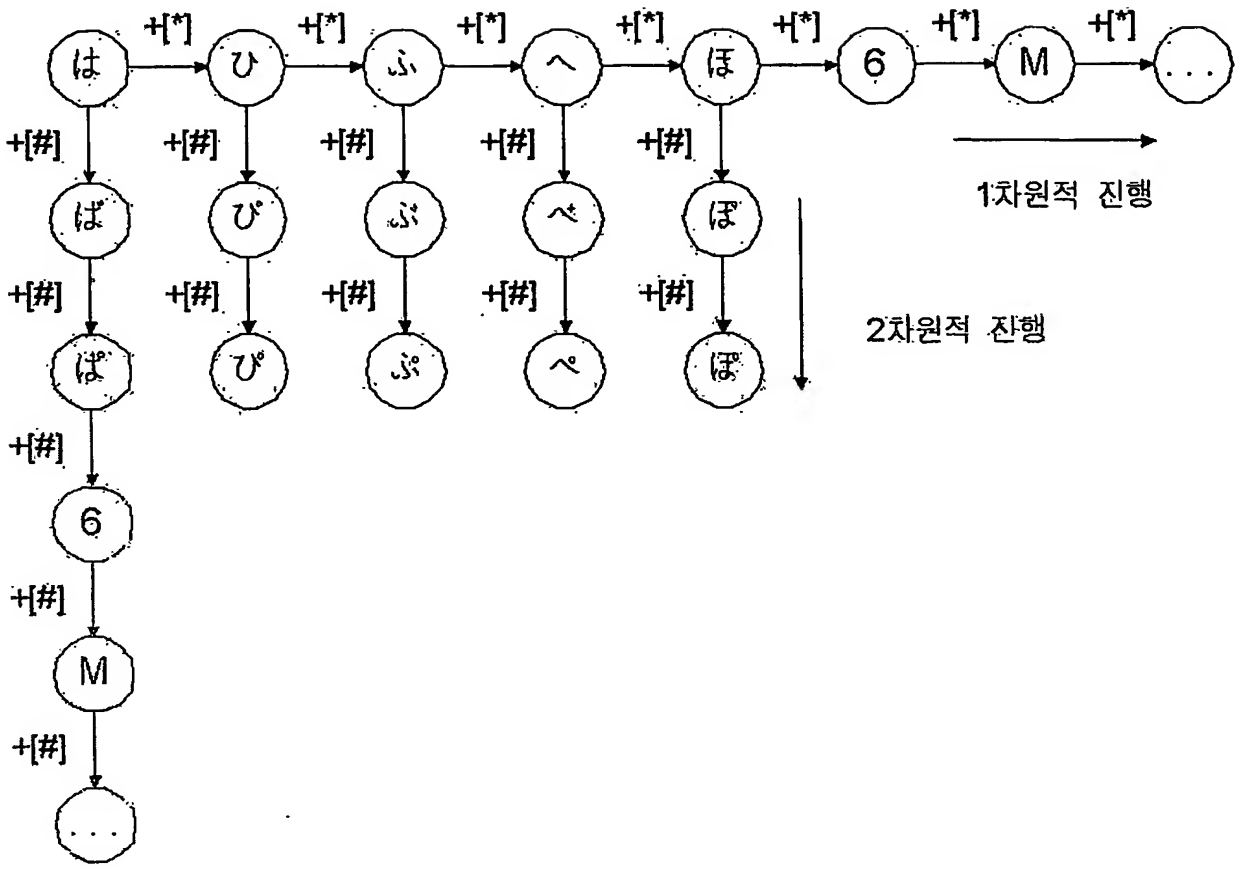

1 あ
. Q Z
2 か
A B C
3 さ
D E F
4 た
G H I
5 な
J K L
6 は
M N O
7 ま
P R S
8 や
T U V
9 ら
W X Y
*
2nd 3rd
0 わ
#
5th 4th

【도 6】

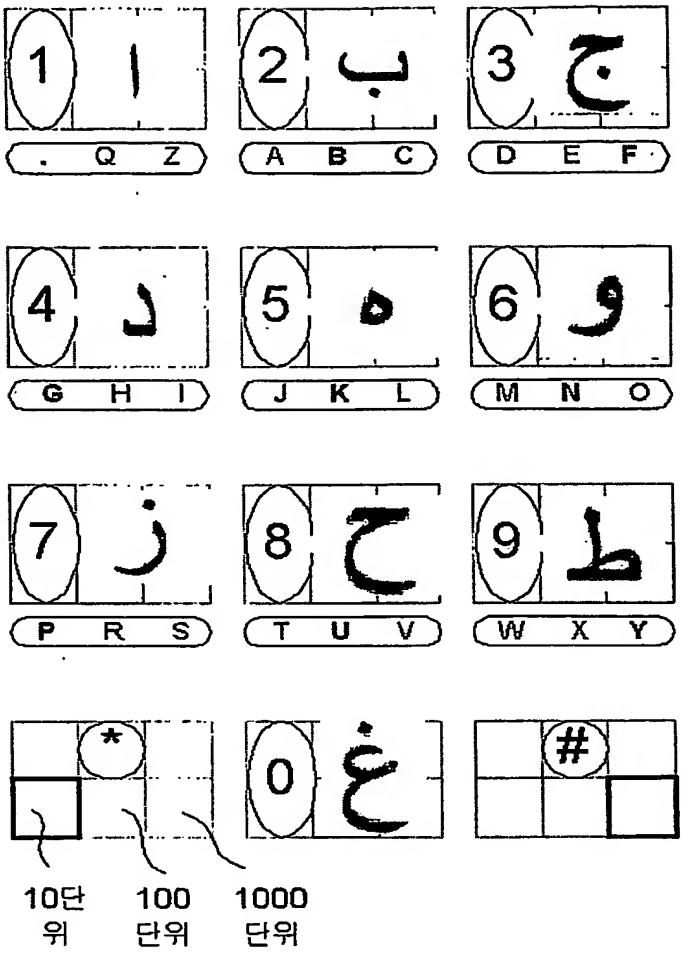
도 2-3



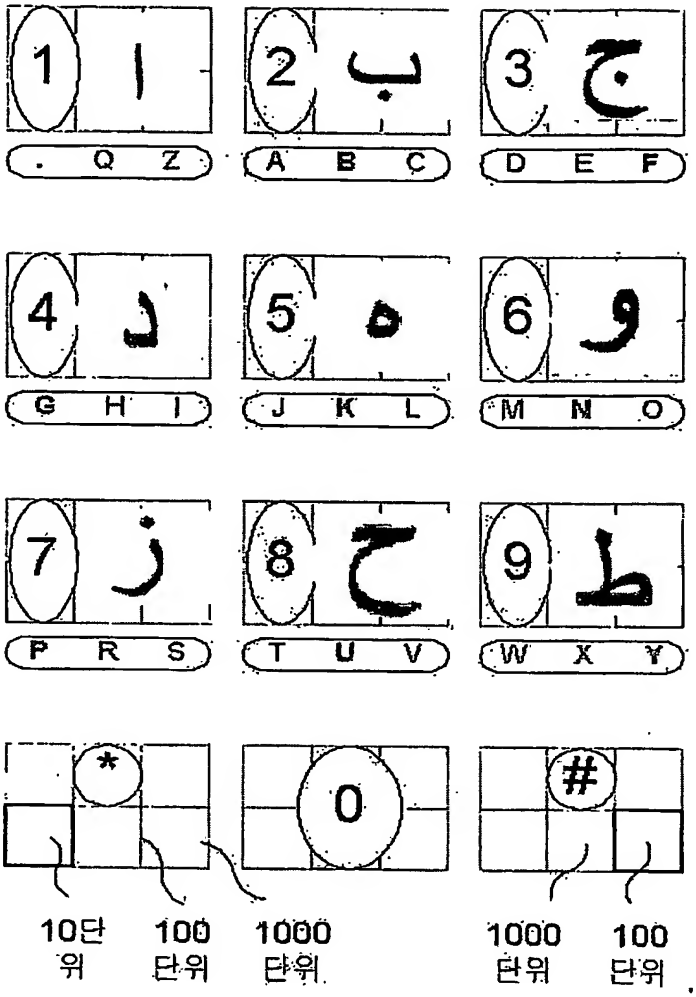
【도 7】
도 2-4



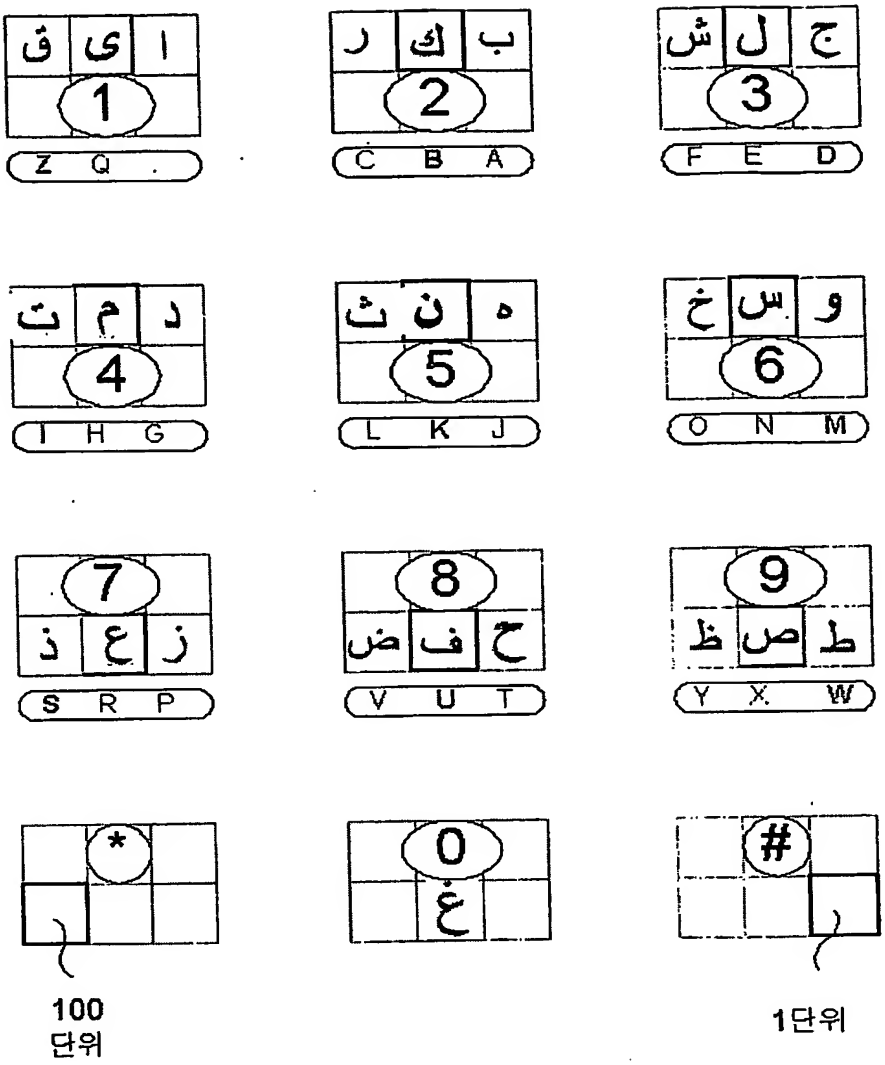
【도 8】
도 3-1


1 ا
. Q Z
2 ب
A B C
3 ج
D E F
4 د
G H I
5 ه
J K L
6 و
M N O
7 ز
P R S
8 ح
T U V
9 ط
W X Y
*
0 غ
#
10단 위
100 단위
1000 단위

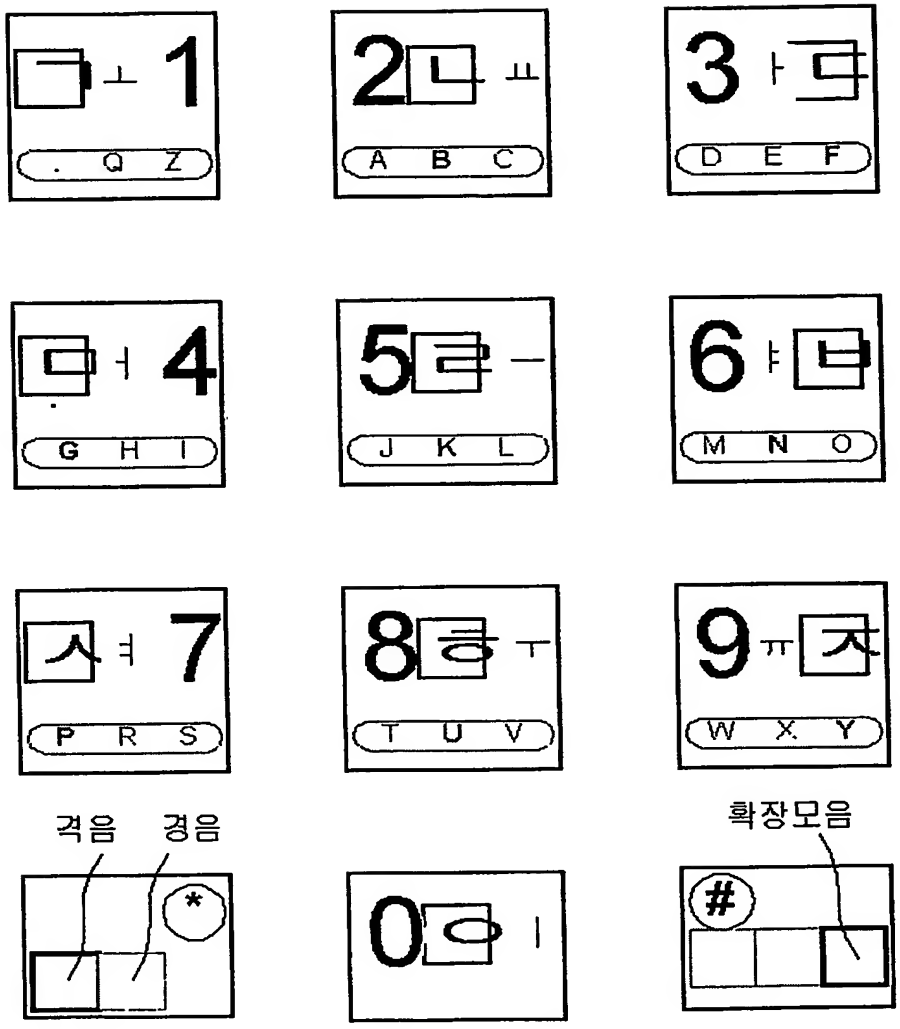
【도 9】
도 3-2


1 ا
. Q Z
2 ب
A B C
3 ج
D E F
4 د
G H I
5 ه
J K L
6 و
M N O
7 ز
P R S
8 ح
T U V
9 ط
W X Y
*
0
#
10단 위
100 단위
1000 단위
1000 단위
100 단위

【도 10】
도 3-3

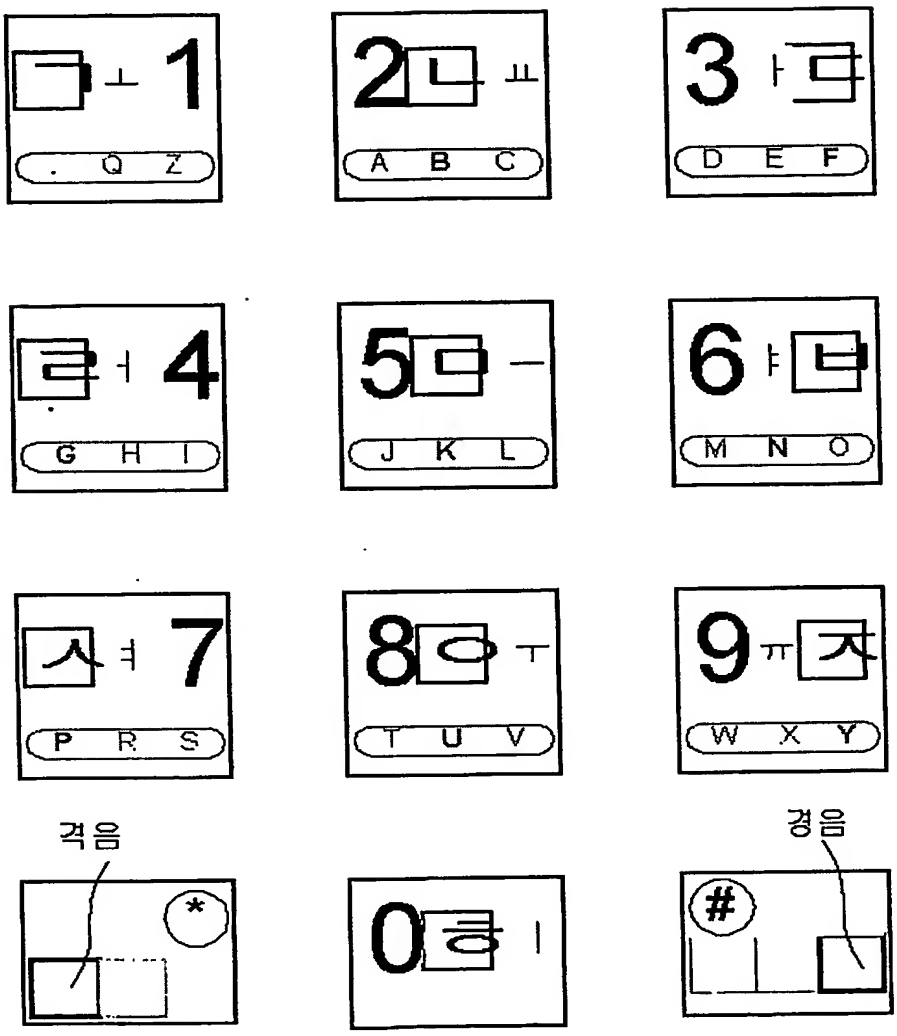

ق ى ا
1
Z Q .
ر ك ب
2
C B A
ش ل ج
3
F E D
ت م د
4
I H G
ث ن ه
5
L K J
خ س و
6
O N M
7
ذ ع ز
S R P
8
ض ف ح
V U T
9
ظ ص ط
Y X W
*
100
단위
0
غ
#
1단위

【도 11】
도 4-1

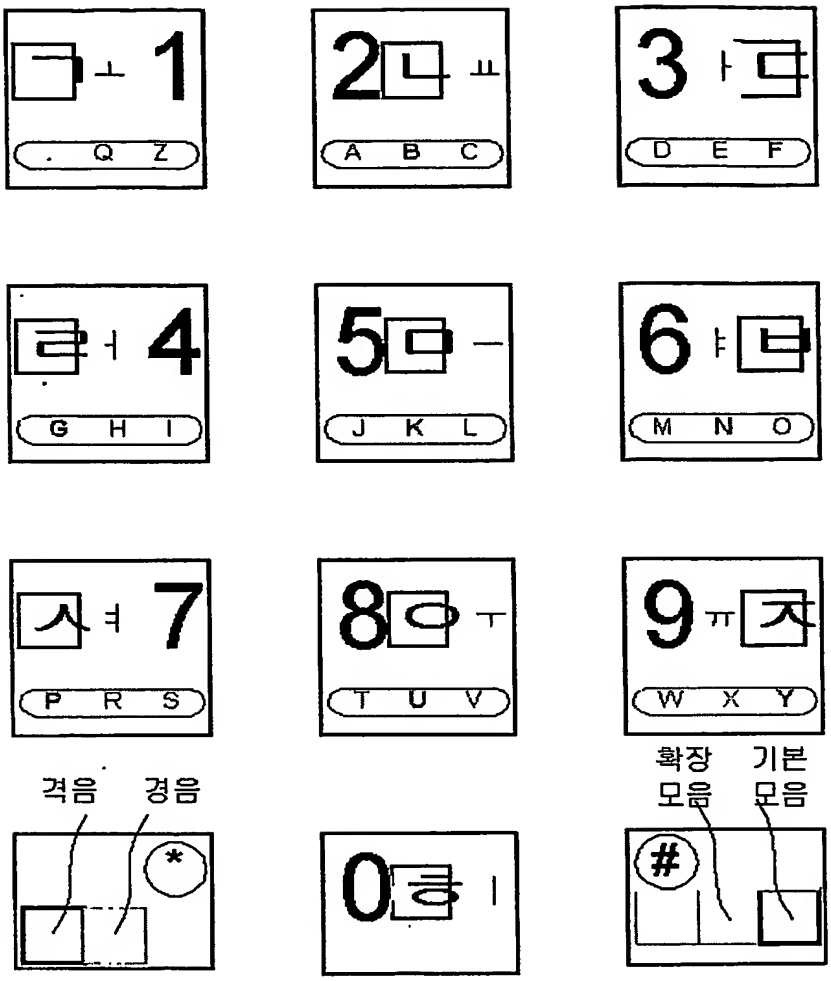


【도 12】

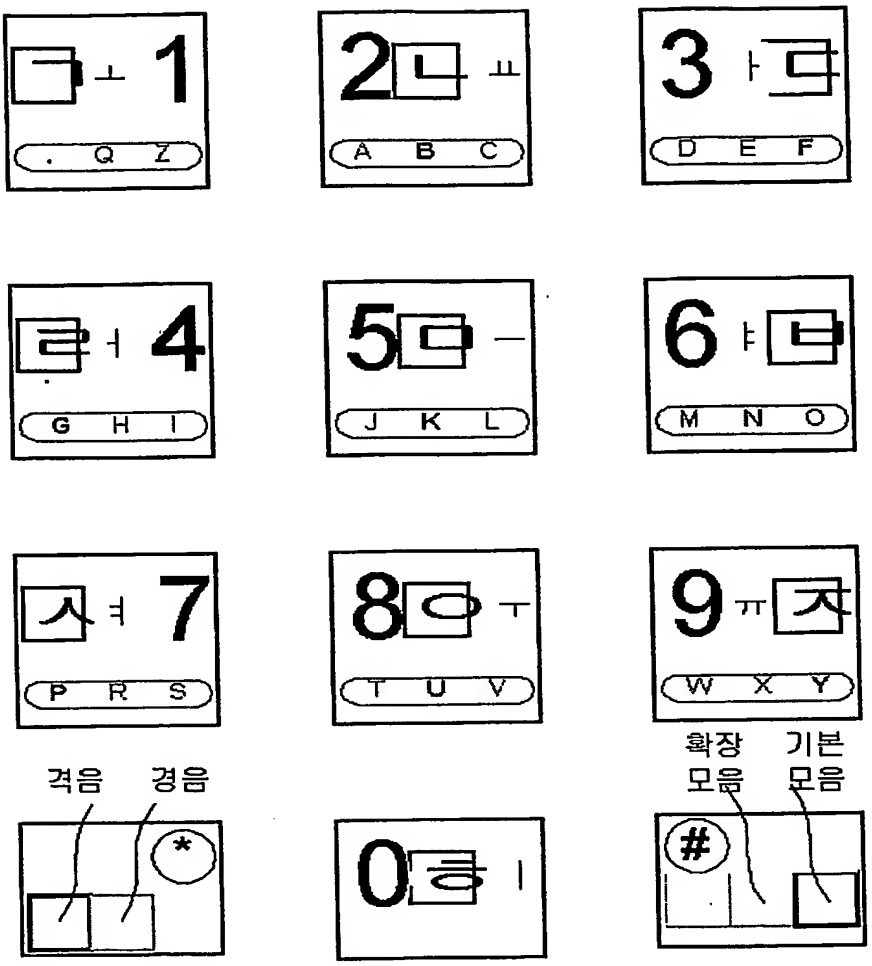
도 4-2



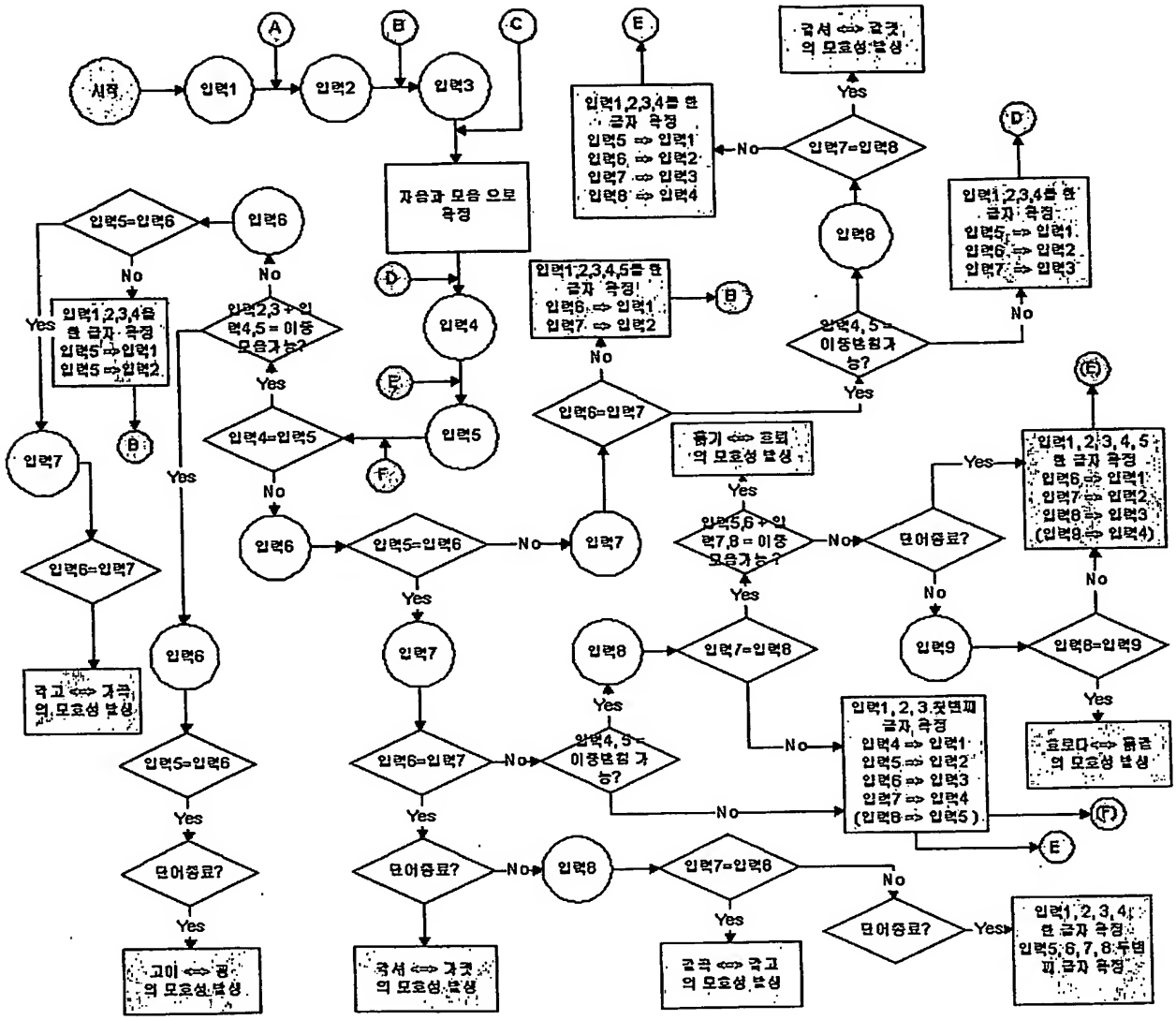
【도 13】
도 4-3



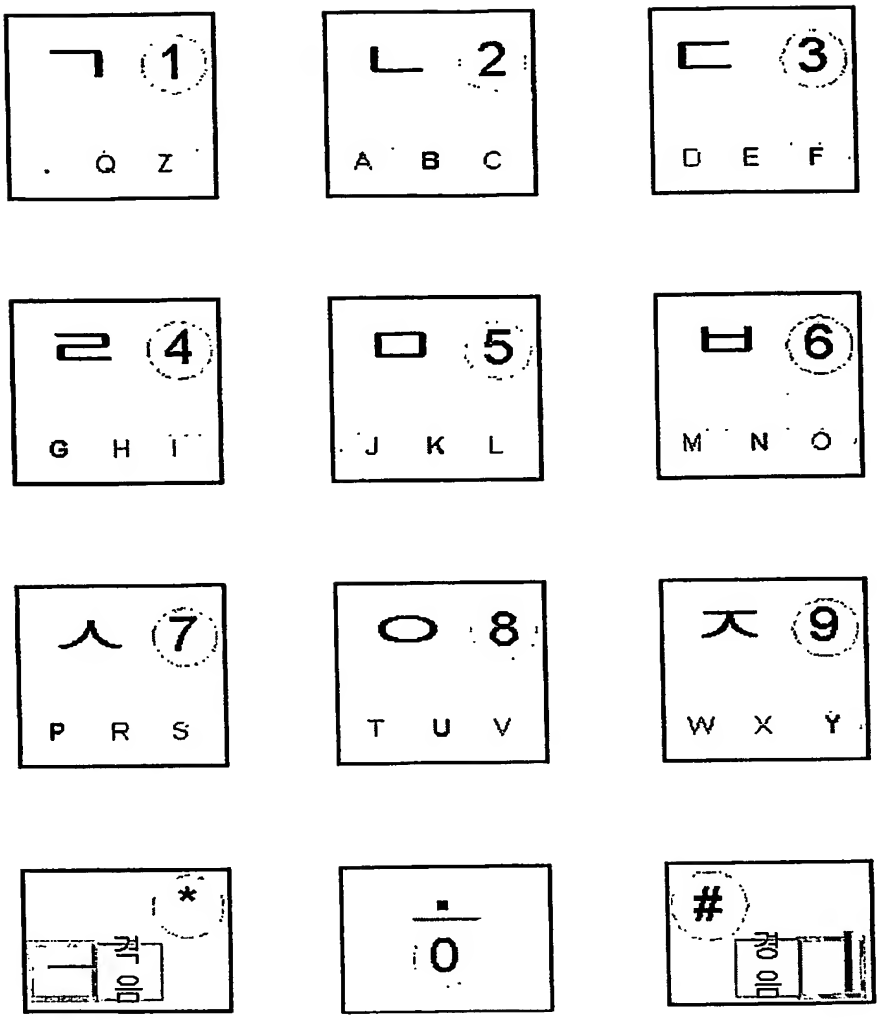
【도 14】
도 4-3



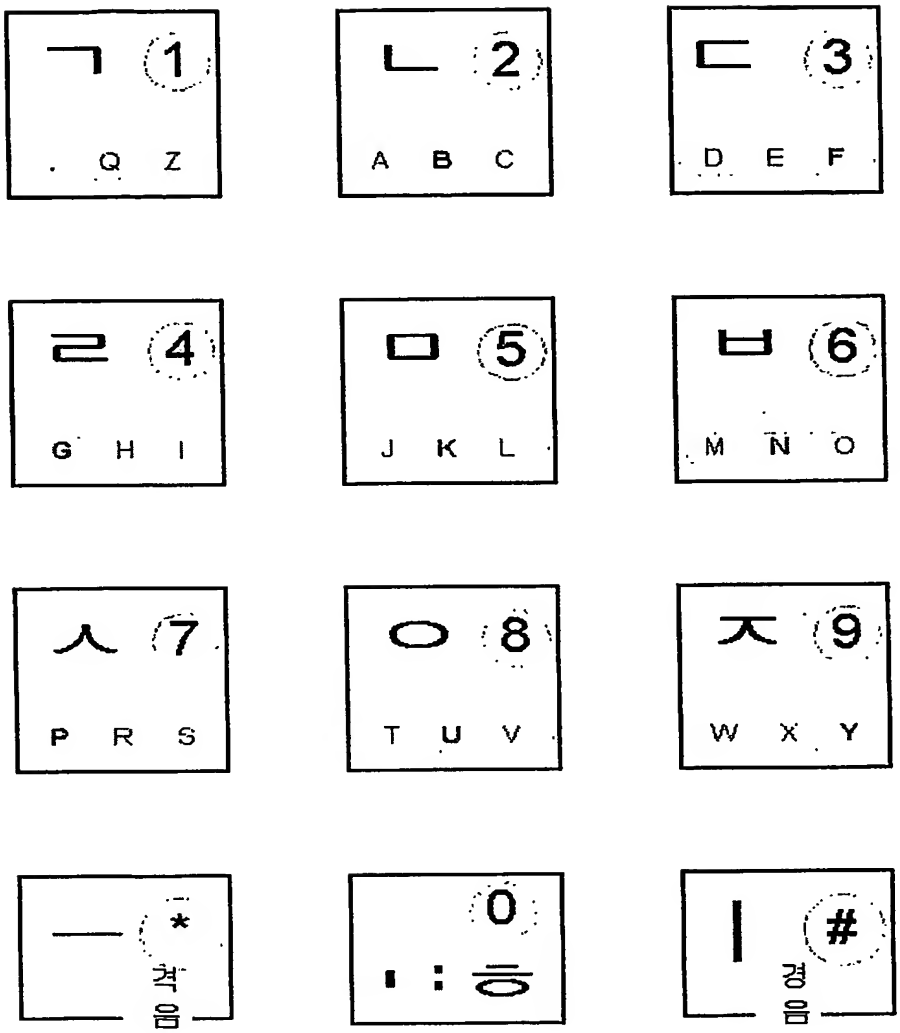
【도 15】
도4-4


시작
입력1
입력2
입력3
자음과 모음 으로 구성
입력4
입력5
입력6
입력7
입력8
입력9
입력5=입력6
입력6=입력7
입력4=입력5
입력7=입력8
입력8=입력9
입력2,3 = 입력4,5 = 이중 모음가능?
입력1,2,3,4를 한 글자 구성 입력5 ⇒ 입력1 입력6 ⇒ 입력2 입력7 ⇒ 입력3 입력8 ⇒ 입력4
입력1,2,3,4,5를 한 글자 구성 입력6 ⇒ 입력1 입력7 ⇒ 입력2
입력1,2,3,4를 한 글자 구성 입력5 ⇒ 입력1 입력6 ⇒ 입력2 입력7 ⇒ 입력3
입력4,5 = 이중분리 가능?
입력5,6 = 입력7,8 = 이중 모음가능?
입력1,2,3,4,5 한 글자 구성 입력6 ⇒ 입력1 입력7 ⇒ 입력2 입력8 ⇒ 입력3 (입력9 ⇒ 입력4)
입력1,2,3 첫번째 글자 구성 입력4 ⇒ 입력1 입력5 ⇒ 입력2 입력6 ⇒ 입력3 입력7 ⇒ 입력4 (입력8 ⇒ 입력5)
입력1,2,3,4 한 글자 구성 입력5,6,7,8 두번째 글자 구성
ㄱㅅ ⇔ ㄱㅈ 의 모호성 발생
ㅎㄱ ⇔ ㅍㅚ 의 모호성 발생
ㄱㄱ ⇔ ㄱㅈ 의 모호성 발생
ㅛㅗㅁ ⇔ ㅠㄱㅈ 의 모호성 발생
ㄱㄱ ⇔ ㄱㄱ 의 모호성 발생
ㄱㅗ ⇔ ㄱㅋ 의 모호성 발생
ㄱㅇ ⇔ ㄱㅎ 의 모호성 발생
ㄱㅅ ⇔ ㄱㅈ 의 모호성 발생
ㄱㄱ ⇔ ㄱㅗ 의 모호성 발생
단어종료?
No
Yes

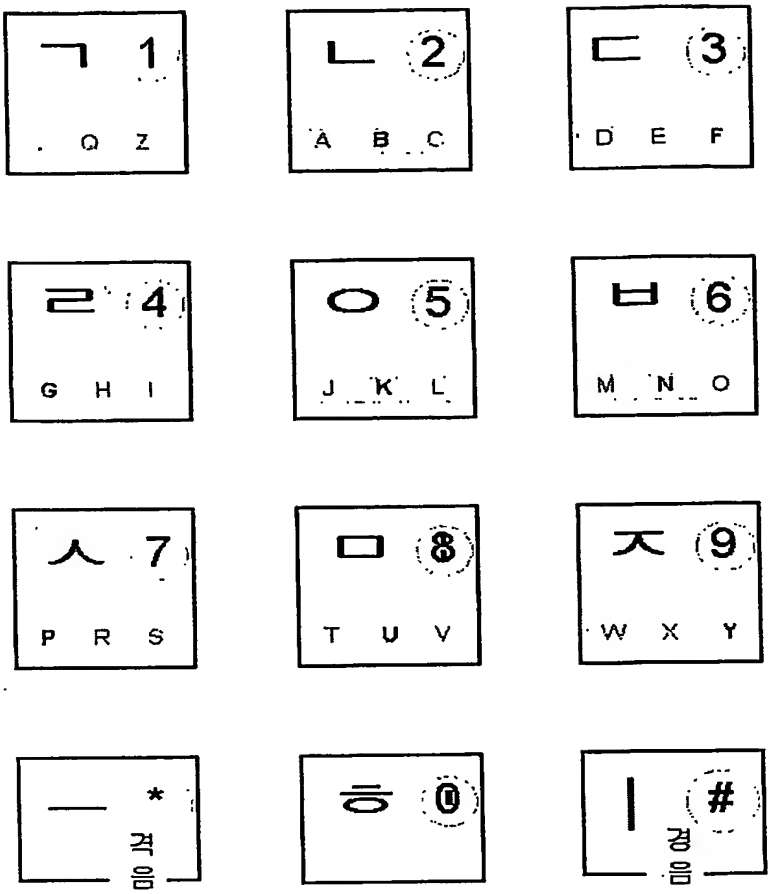
【도 16】
도 4-5



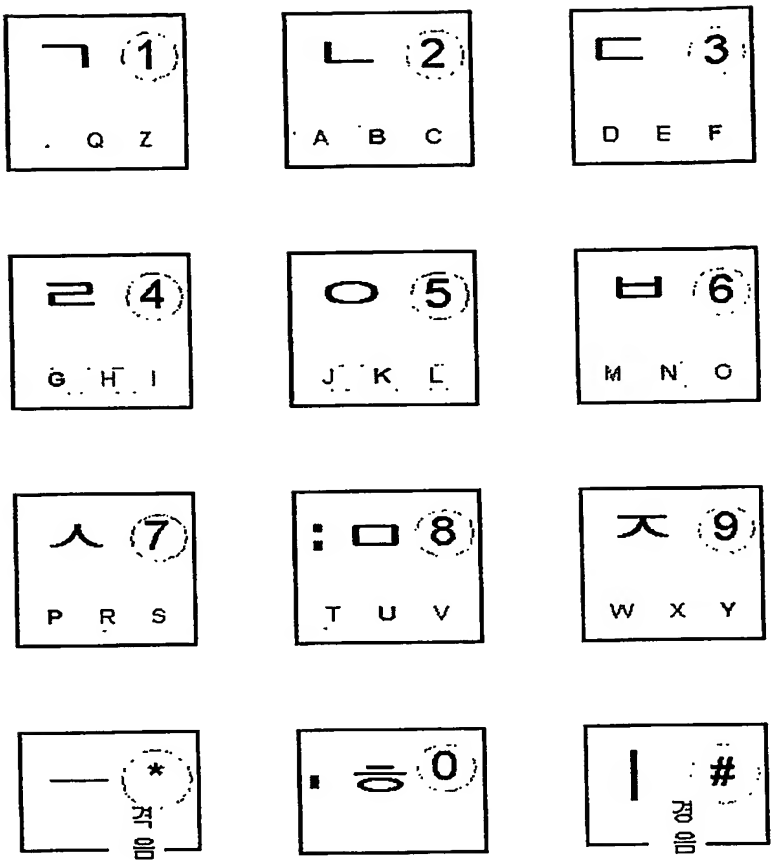
【도 17】
도 4-6


ㄱ 1
. Q Z
ㄴ 2
A B C
ㄷ 3
D E F
ㄹ 4
G H I
ㅁ 5
J K L
ㅂ 6
M N O
ㅅ 7
P R S
ㅇ 8
T U V
ㅈ 9
W X Y
— *
격
음
0
ㅣ : ㅎ
| #
경
음

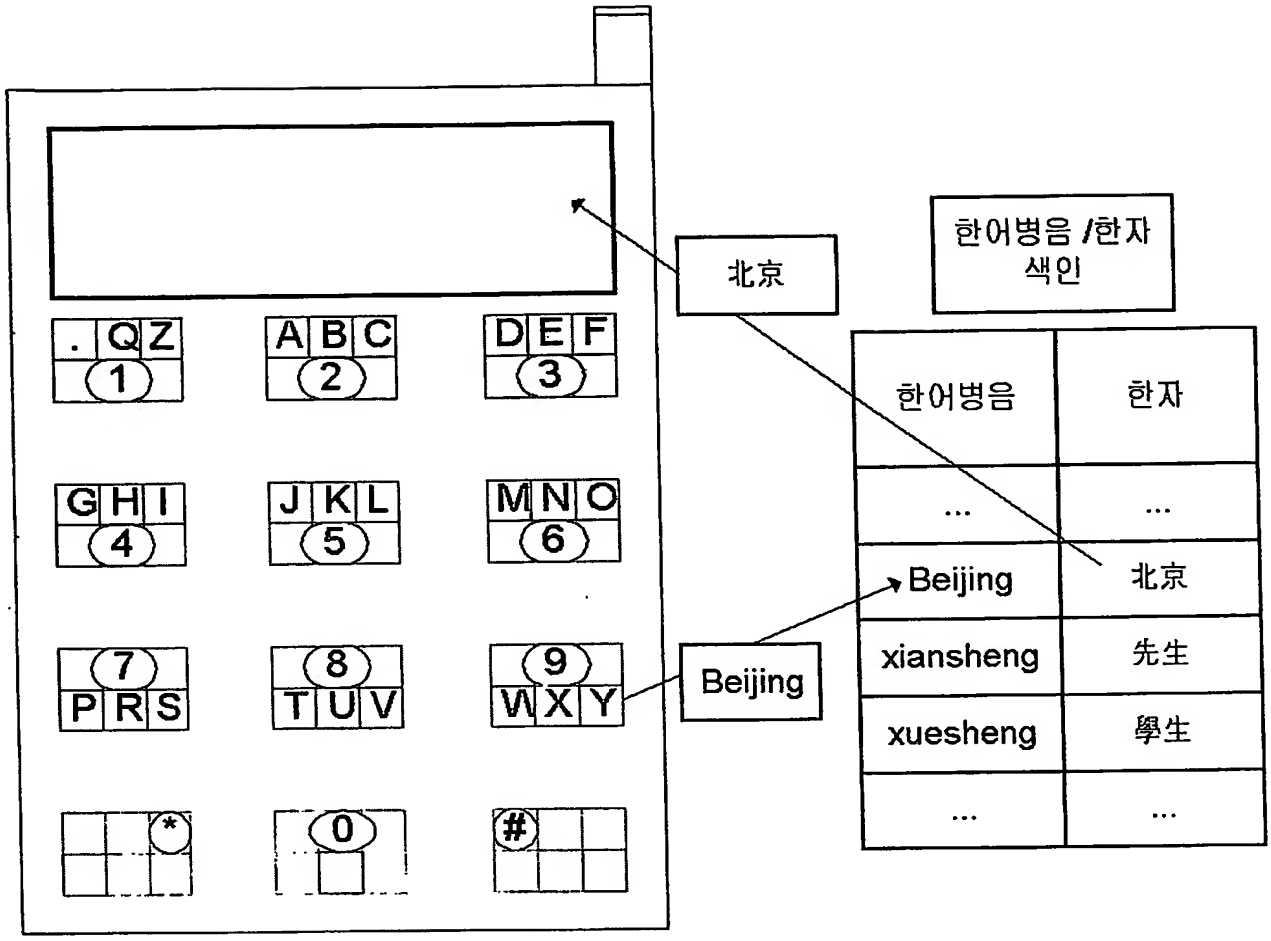
【도 18】
도 4-7


ㄱ 1
. Q Z
ㄴ 2
A B C
ㄷ 3
D E F
ㄹ 4
G H I
ㅇ 5
J K L
ㅂ 6
M N O
ㅅ 7
P R S
ㅁ 8
T U V
ㅈ 9
W X Y
— *
격
음
ㅎ 0
ㅣ #
경
음

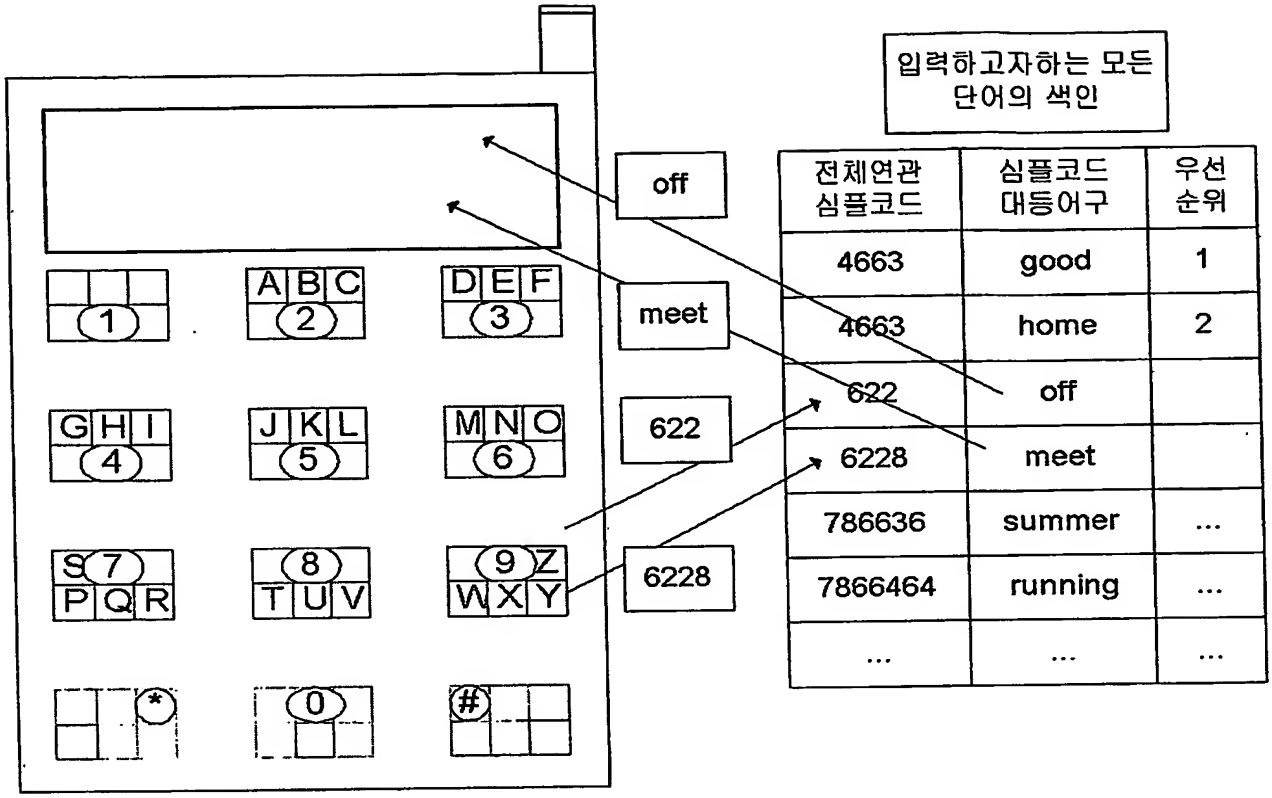
【도 19】
도 4-8


ㄱ 1
. Q Z
ㄴ 2
A B C
ㄷ 3
D E F
ㄹ 4
G H I
ㅇ 5
J K L
ㅂ 6
M N O
ㅅ 7
P R S
: ㅁ 8
T U V
ㅈ 9
W X Y
— *
격
음
: ㅎ 0
| #
경
음

【도 20】
도 5-1

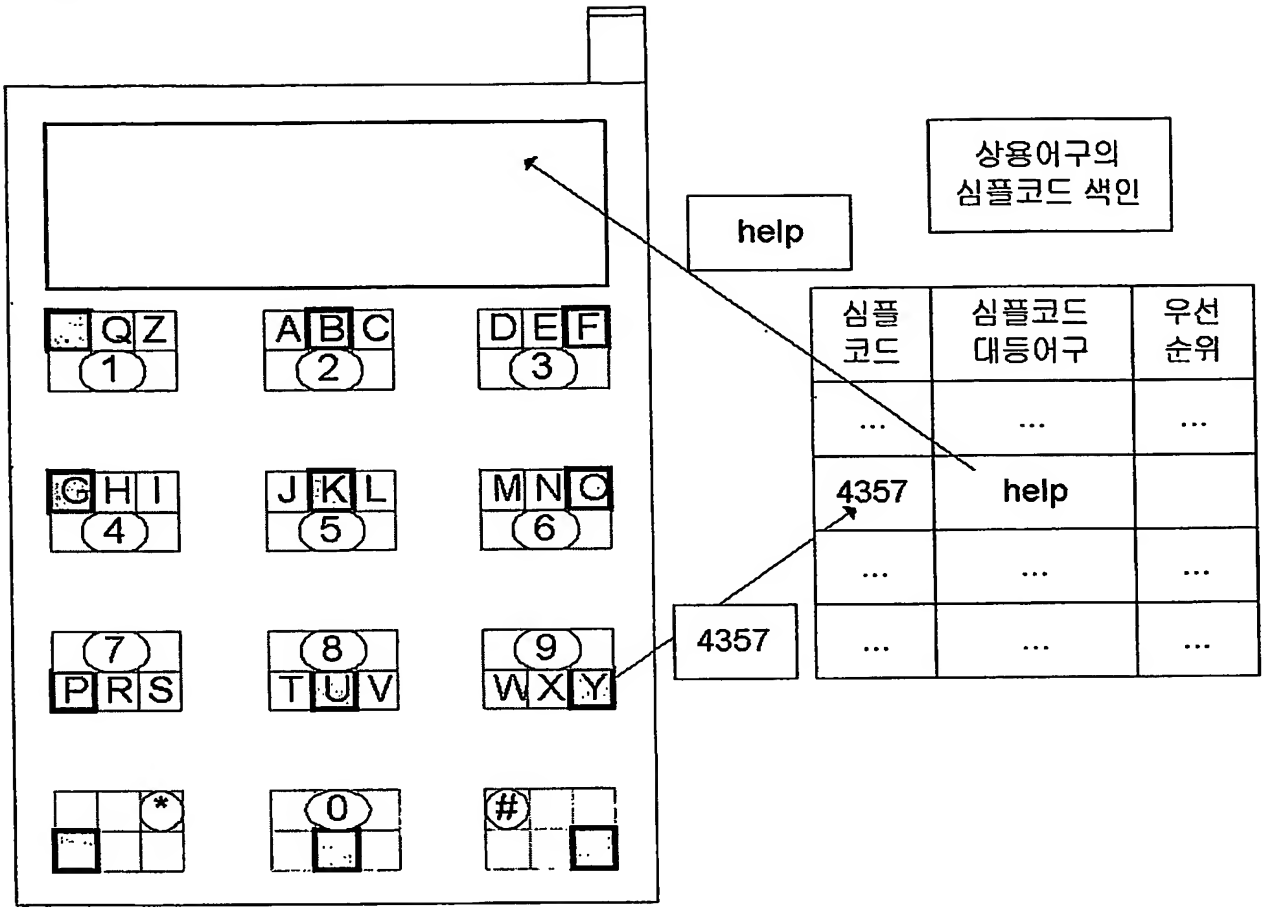

北京
한어병음 /한자
색인
한어병음
한자
...
...
Beijing
北京
xiansheng
先生
xuesheng
學生
...
...
Beijing
. Q Z
1
A B C
2
D E F
3
G H I
4
J K L
5
M N O
6
7
P R S
8
T U V
9
W X Y
*
0
#

【도 21】
도 5-2

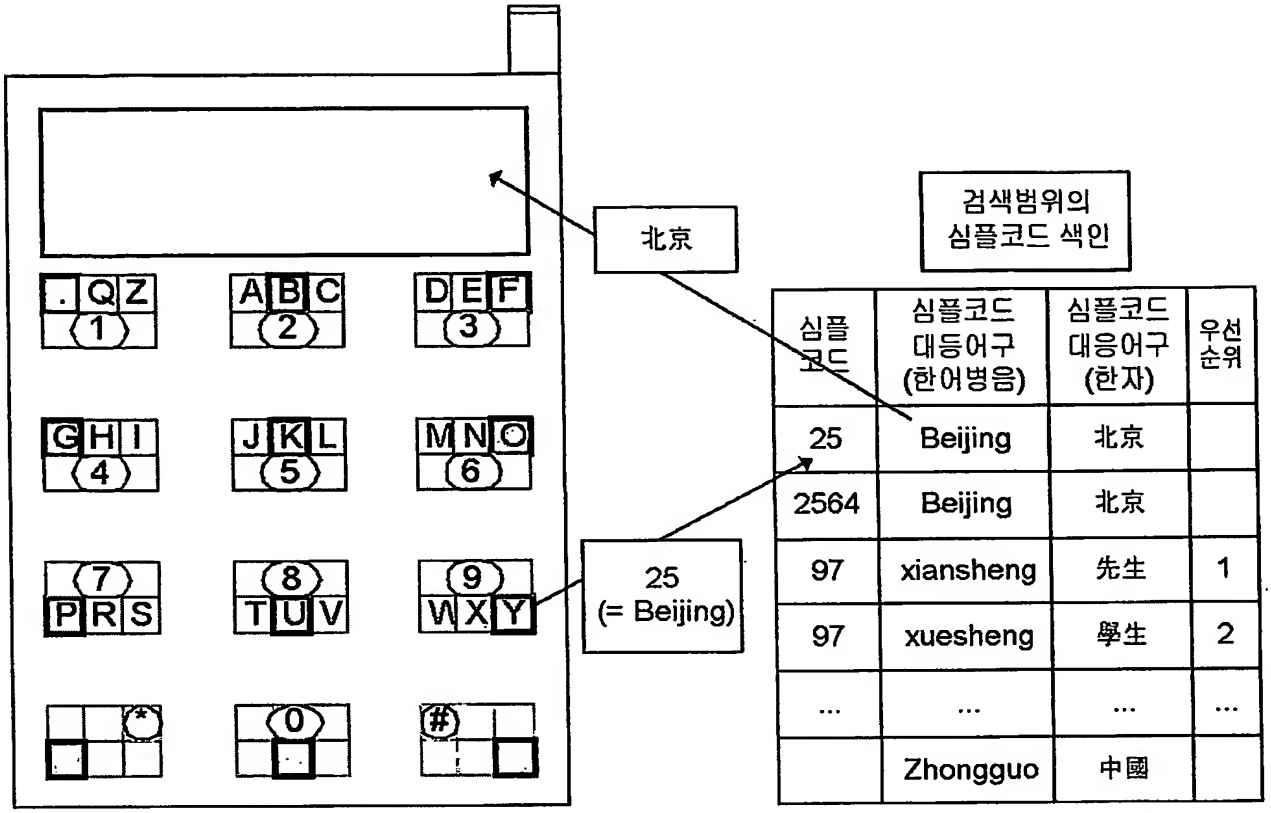


전체연관 심플코드	심플코드 대등어구	우선 순위
4663	good	1
4663	home	2
622	off	
6228	meet	
786636	summer	...
7866464	running	...
...	...	...

【도 22】
도 5-3

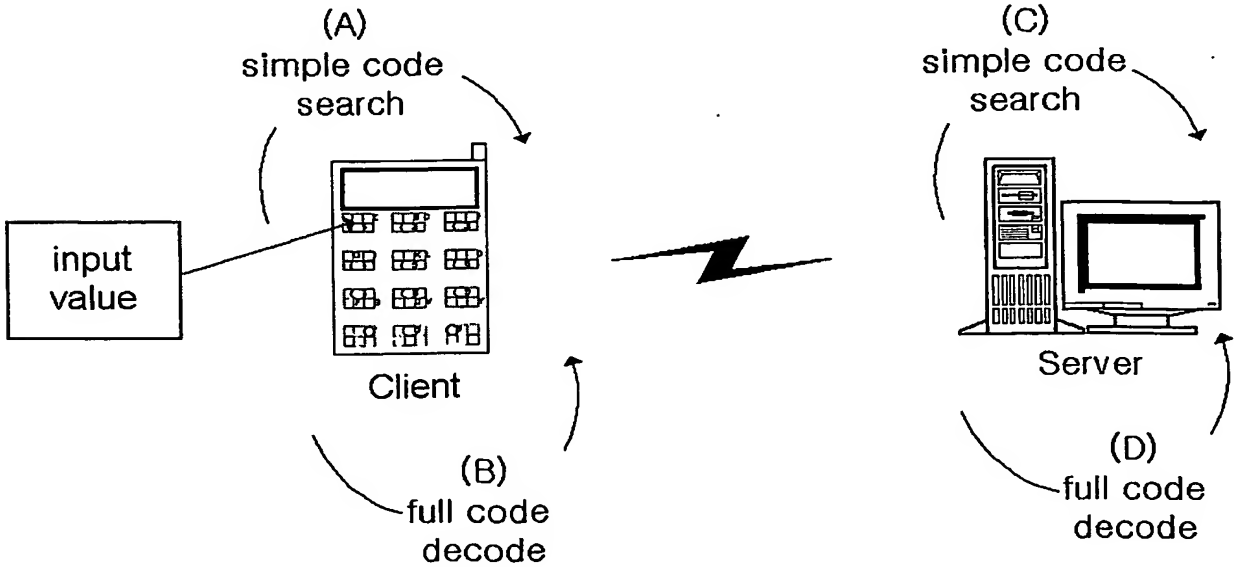

상용어구의
심플코드 색인
help
심플
코드
심플코드
대등어구
우선
순위
...
4357
help
4357
Q Z
1
A B C
2
D E F
3
G H I
4
J K L
5
M N O
6
7
P R S
8
T U V
9
W X Y
*
0
#

【도 23】
도 5-4

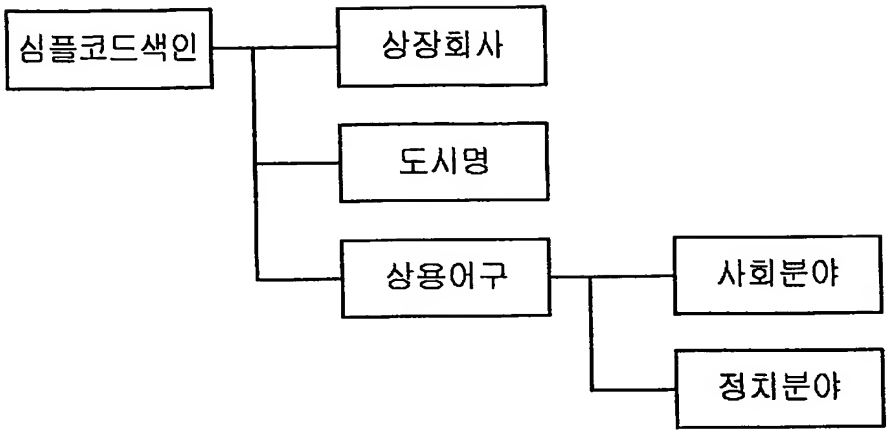


심플코드	심플코드 대응어구 (한어병음)	심플코드 대응어구 (한자)	우선순위
25	Beijing	北京	
2564	Beijing	北京	
97	xiansheng	先生	1
97	xuesheng	學生	2
...	...	...	...
	Zhongguo	中國	

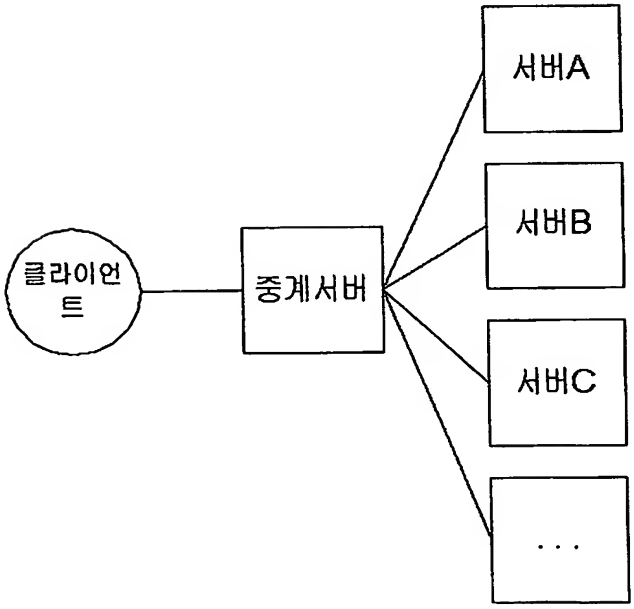
【도 24】
도 5-5



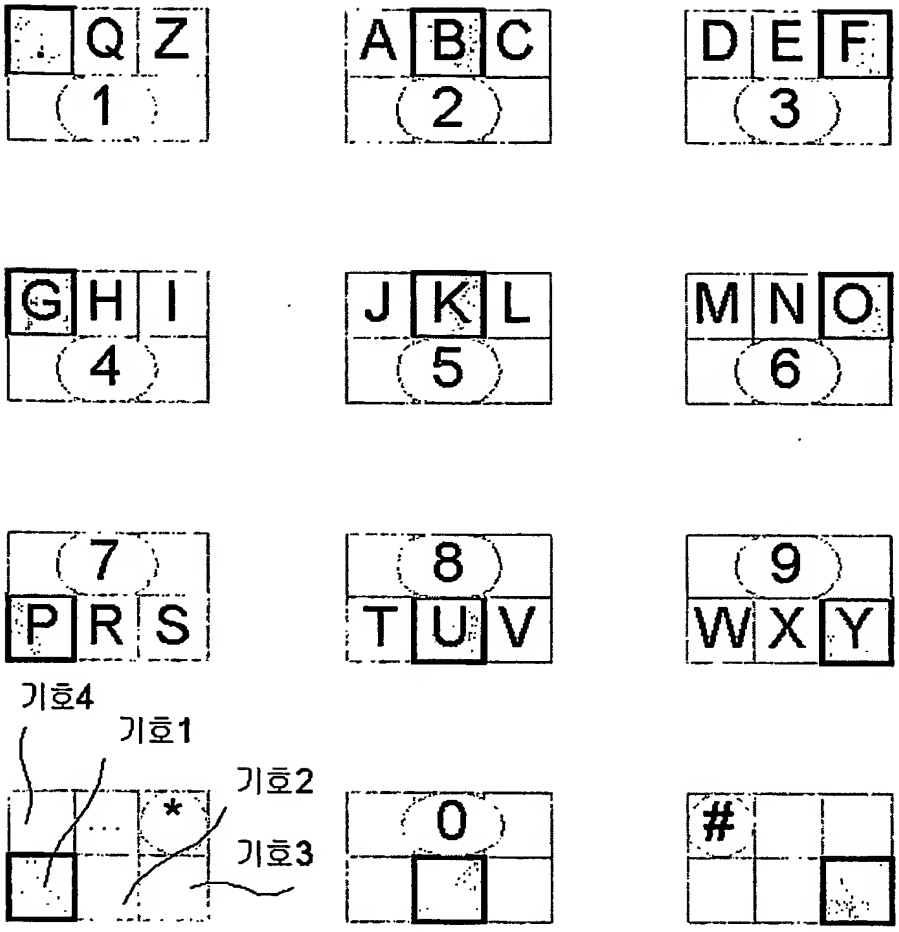
【도 25】
도 5-6



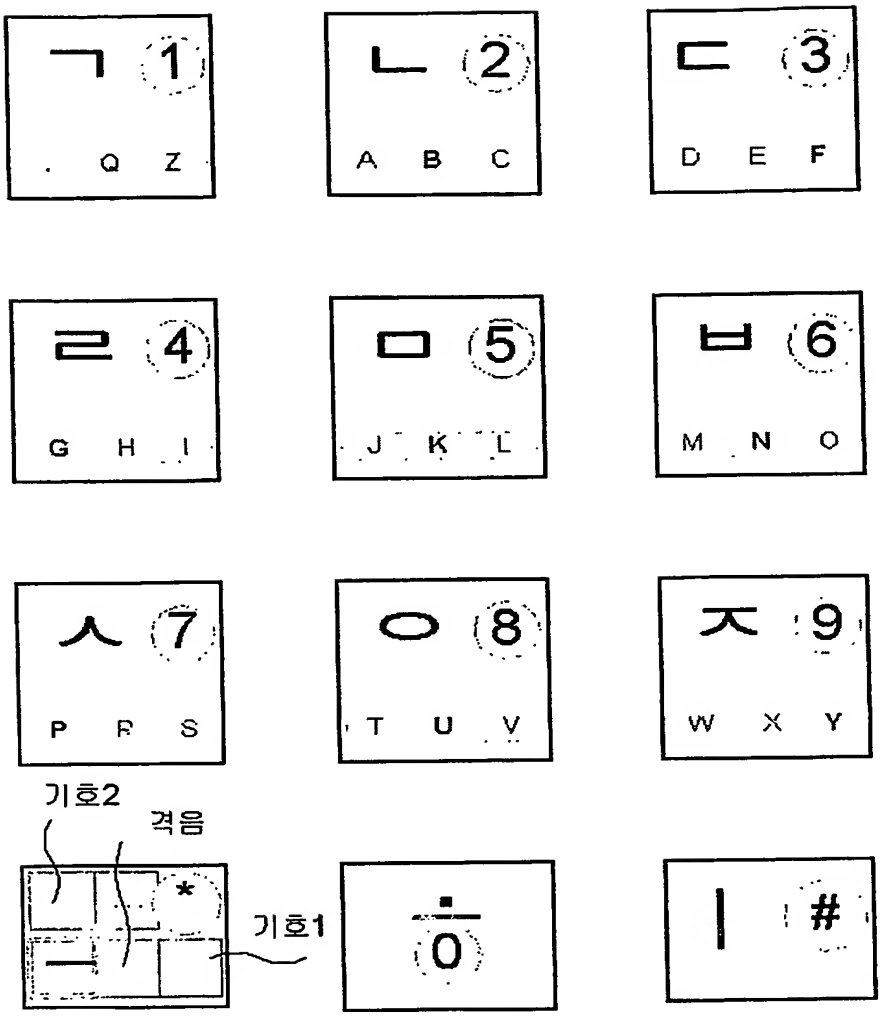
【도 26】
도 6-1



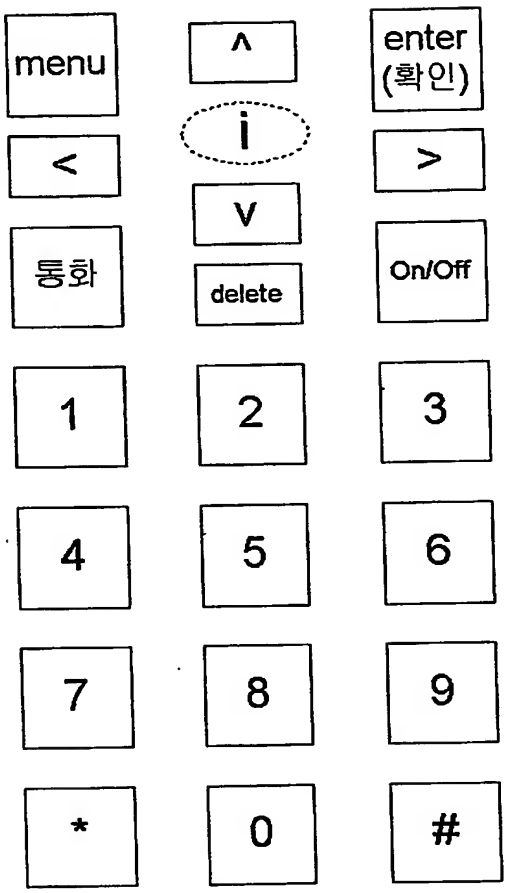
【도 27】
도 7-1



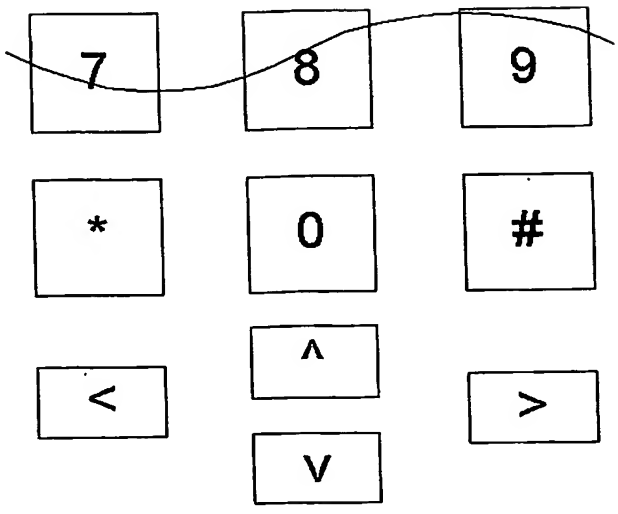
【도 28】
도 7-2


ㄱ 1
. Q Z
ㄴ 2
A B C
ㄷ 3
D E F
ㄹ 4
G H I
ㅁ 5
J K L
ㅂ 6
M N O
ㅅ 7
P R S
ㅇ 8
T U V
ㅈ 9
W X Y
기호2
격음
*
기호1
0
#

【도 29】
도 8-1

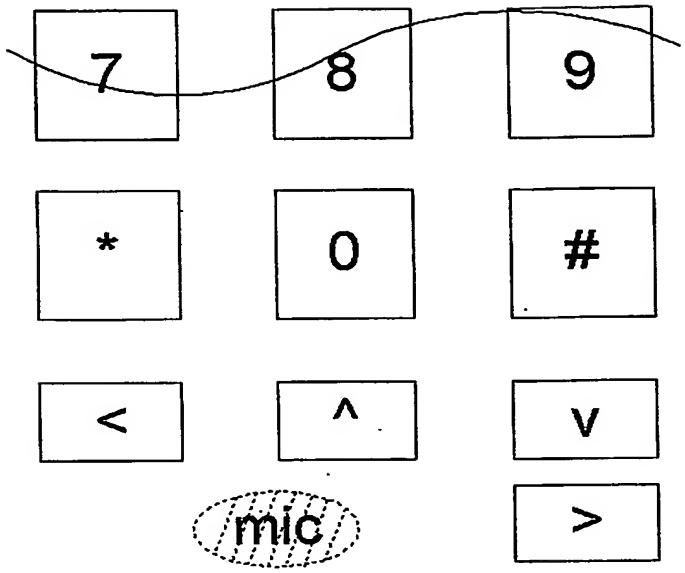


【도 30】
도 8-2



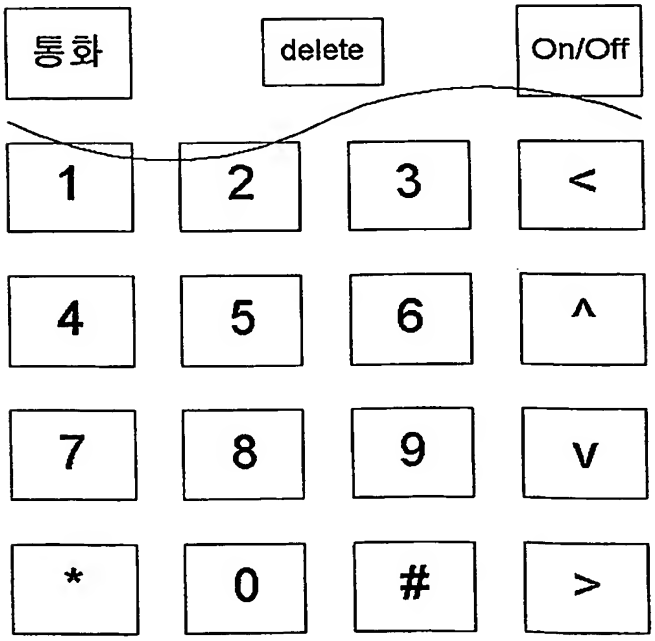
【도 31】

도 8-3

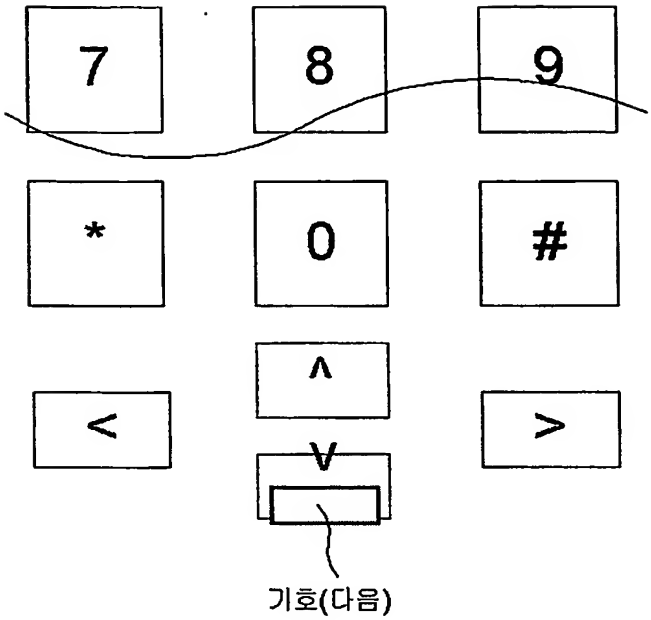


【도 32】

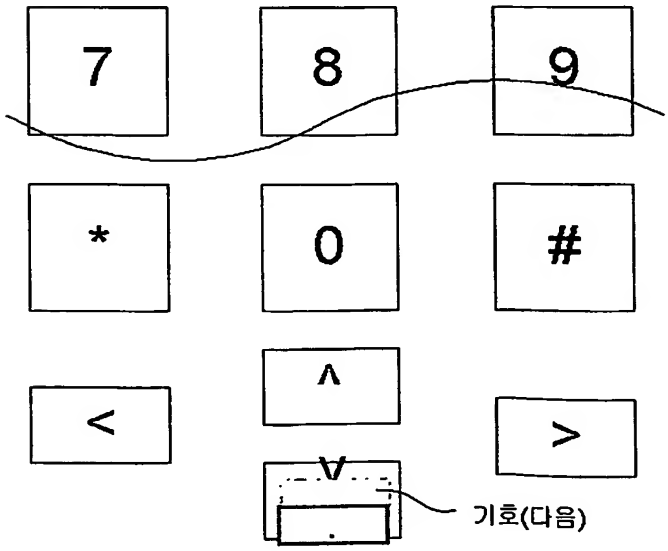
도 8-4



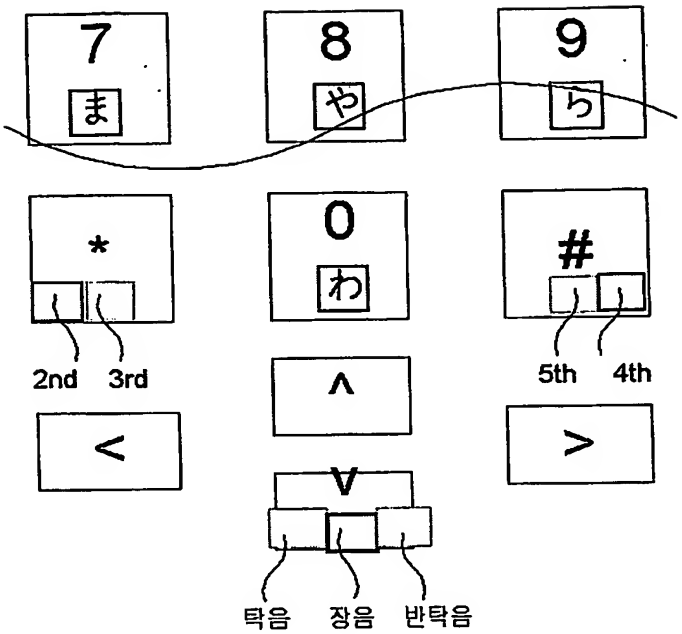
【도 33】
도 8-5



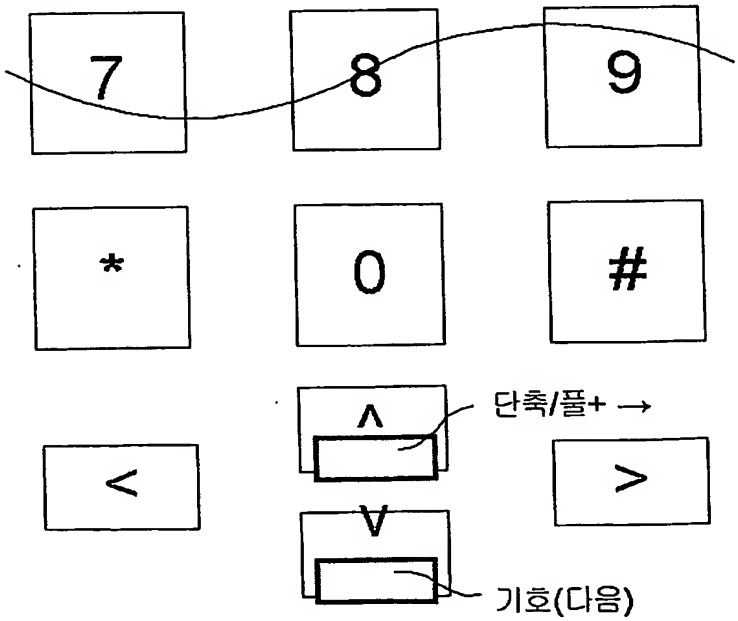
【도 34】
도 8-6



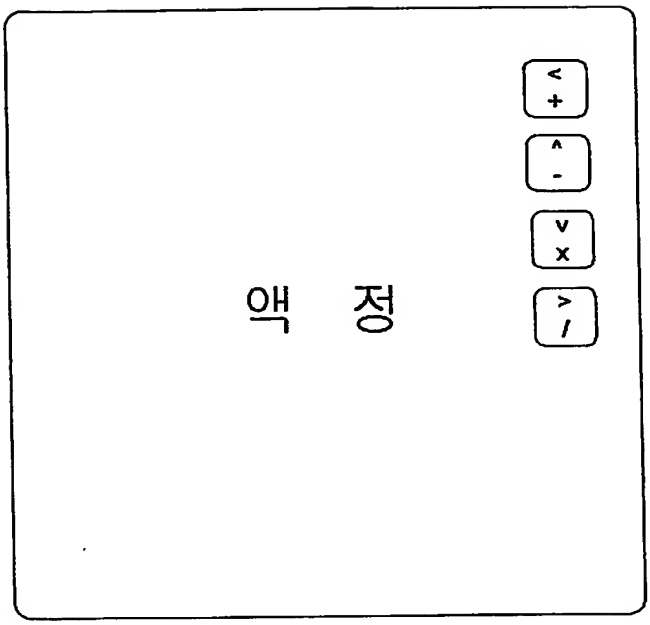
【도 35】
도 8-7



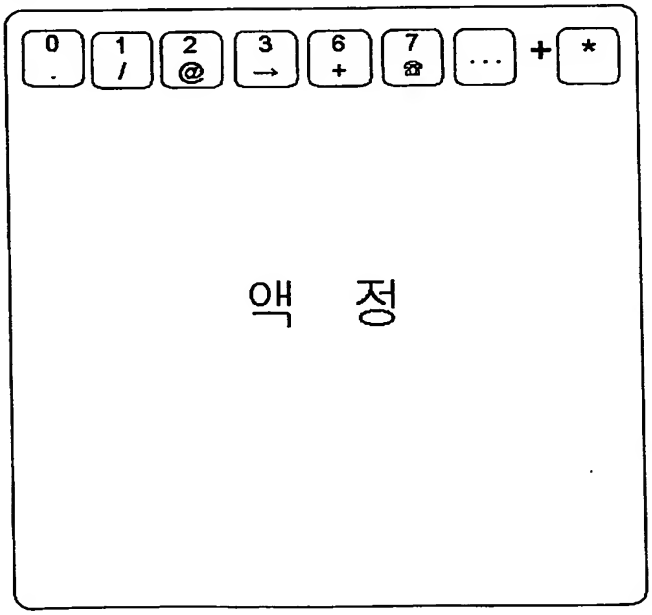
【도 36】
도 8-8



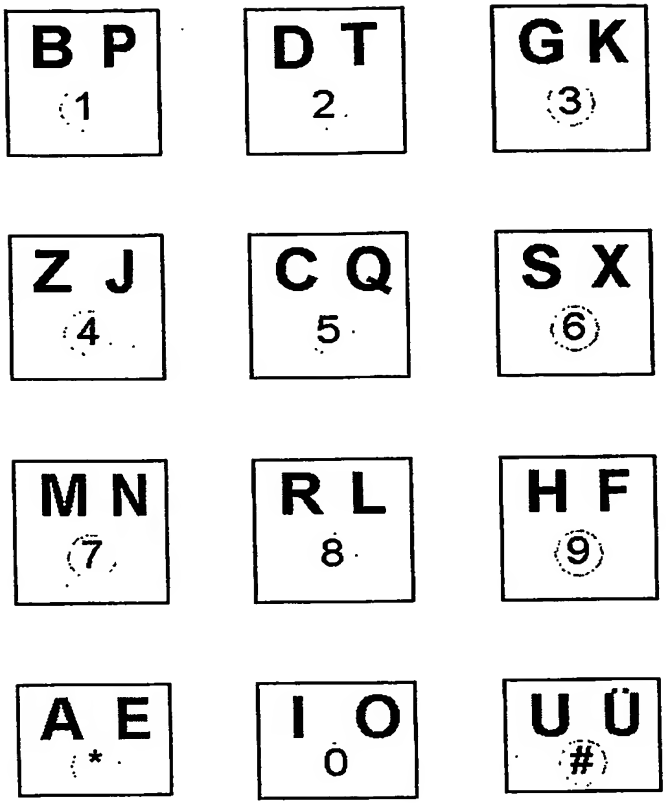
【도 37】
도 9-1



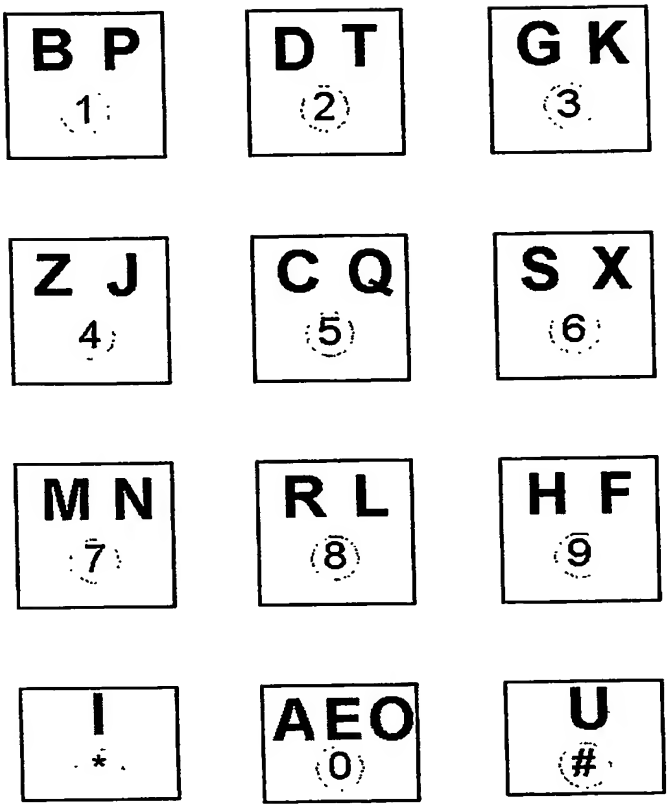
【도 38】
도 9-2



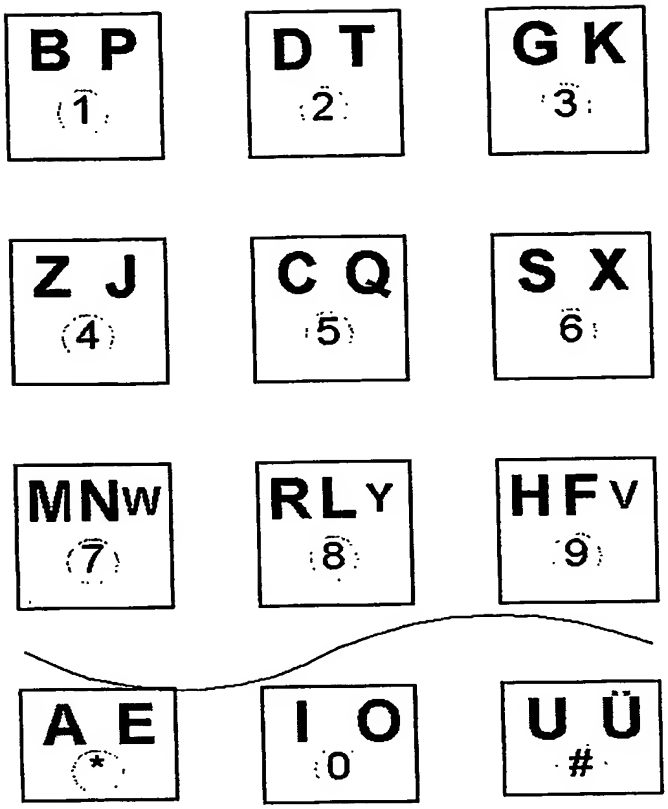
【도 39】
도 10-1



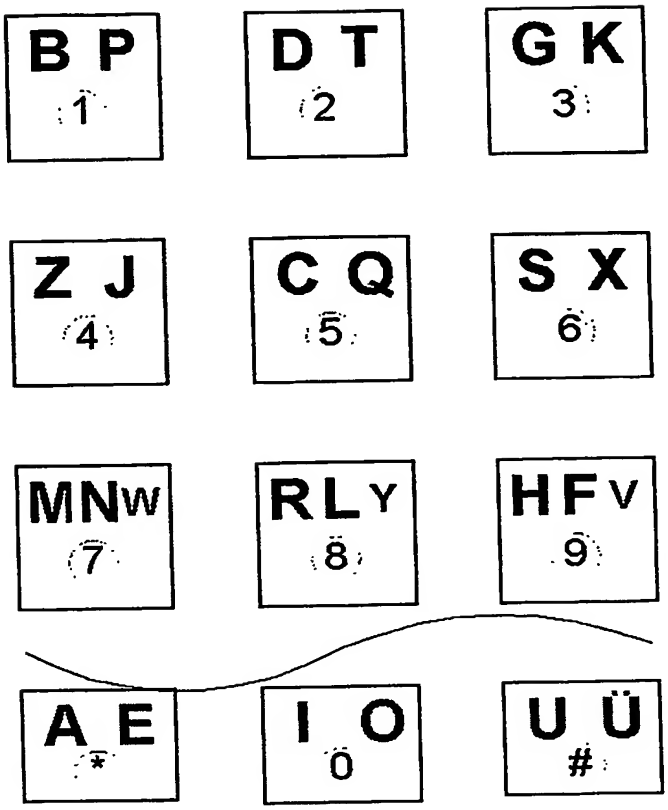
【도 40】
도 10-2



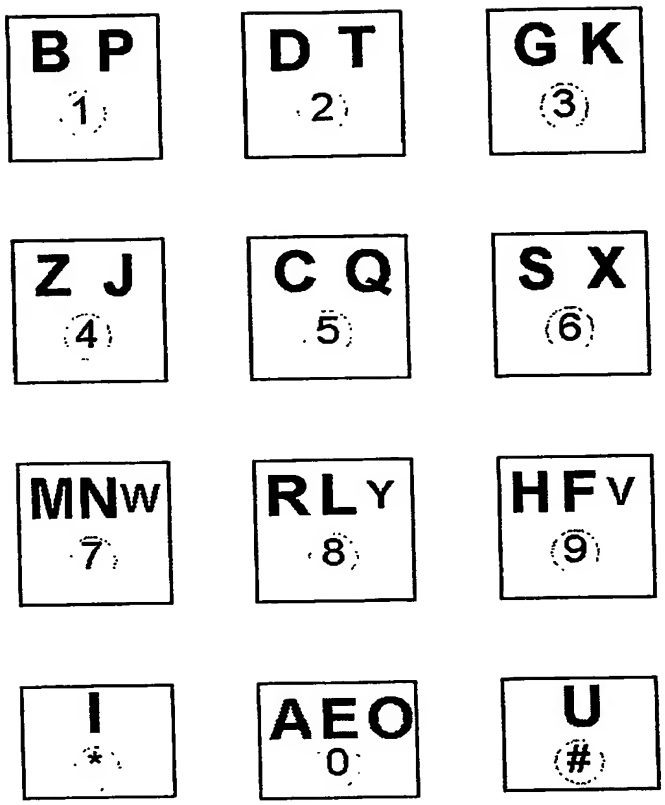
【도 41】
도 10-3



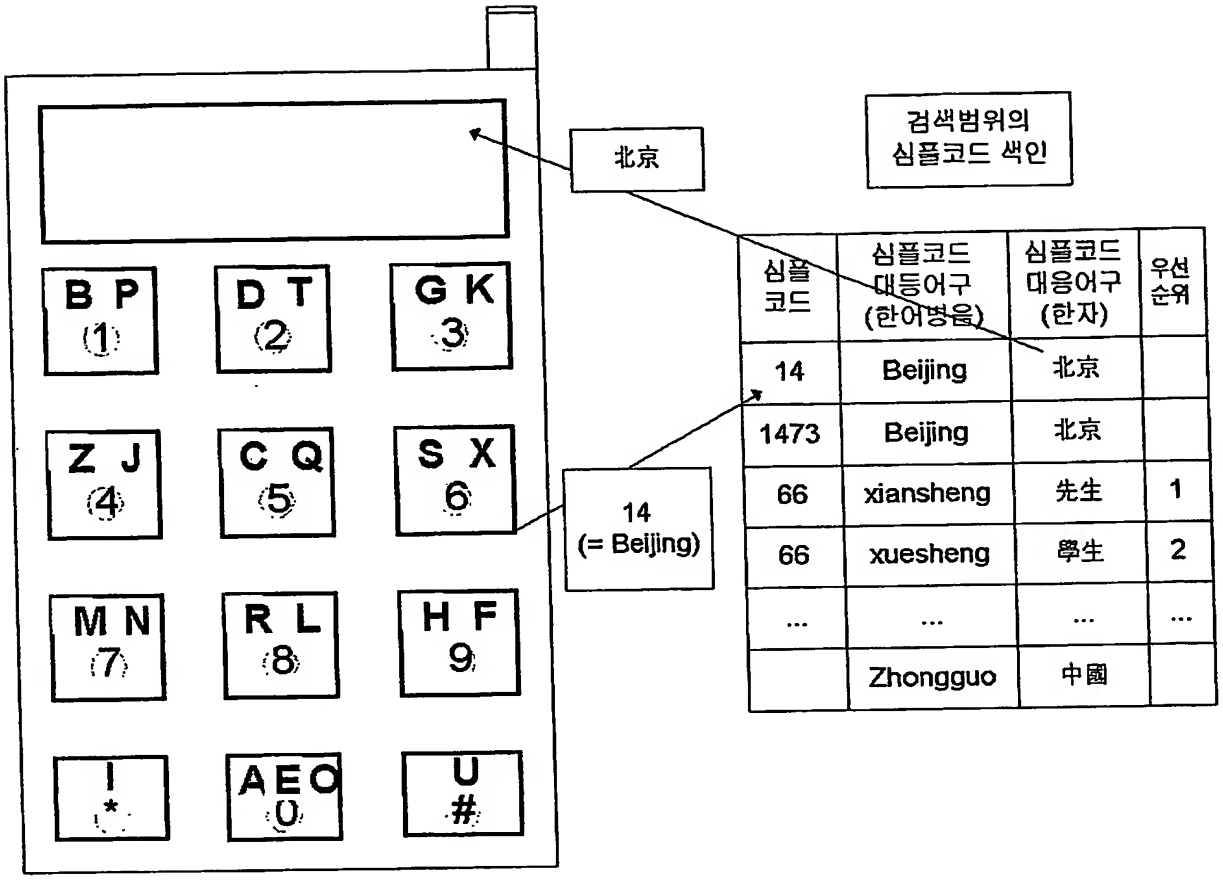
【도 42】
도 10-3



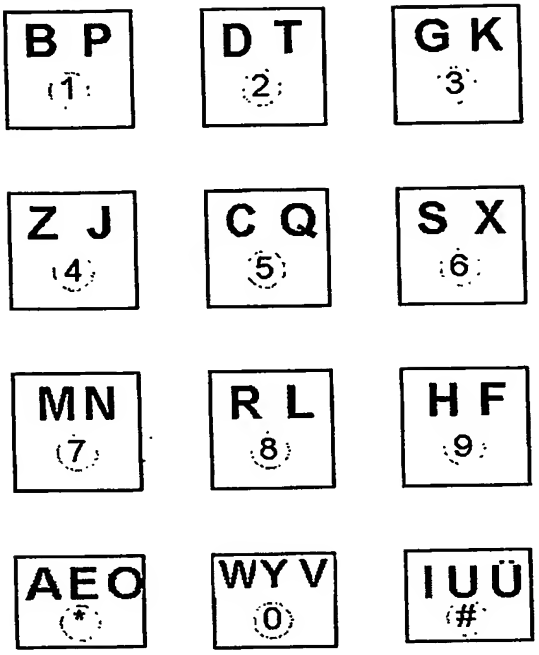
【도 43】
도 10-4



【도 44】
도 10-5

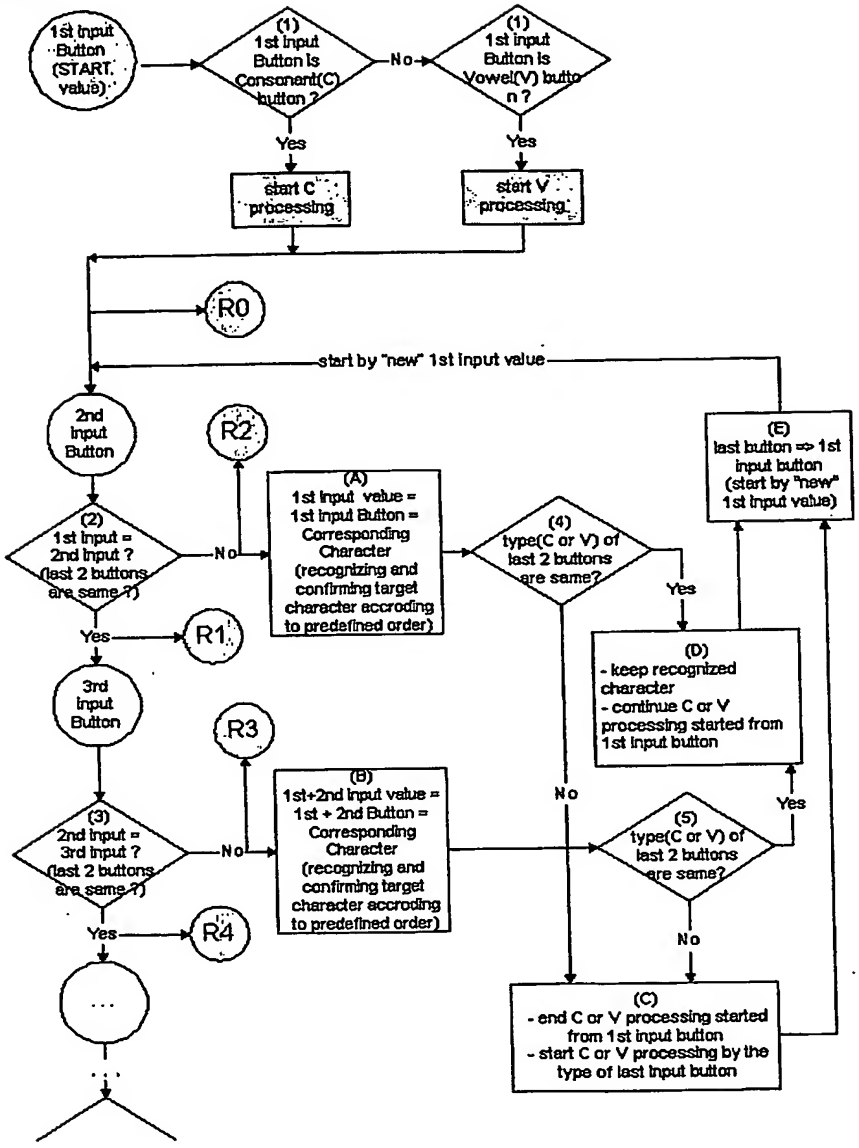

B P (1)
D T (2)
G K (3)
Z J (4)
C Q (5)
S X (6)
M N (7)
R L (8)
H F (9)
I *
A E O (0)
U #
北京
14 (= Beijing)
검색범위의 심플코드 색인
심플 코드
심플코드 대등어구 (한어병음)
심플코드 대응어구 (한자)
우선 순위
14
Beijing
北京
1473
Beijing
北京
66
xiansheng
先生
1
66
xuesheng
學生
2
...
...
...
...
Zhongguo
中國

【도 55】
도 10-6



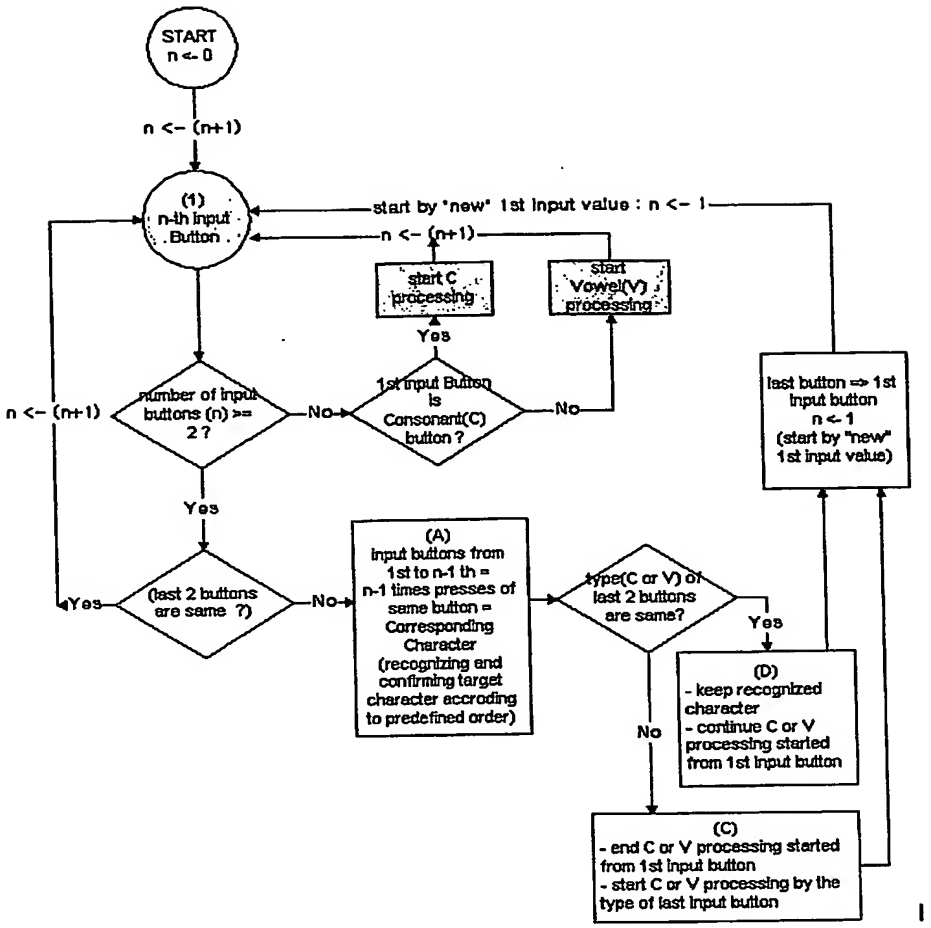
【도 56】

도 10-7

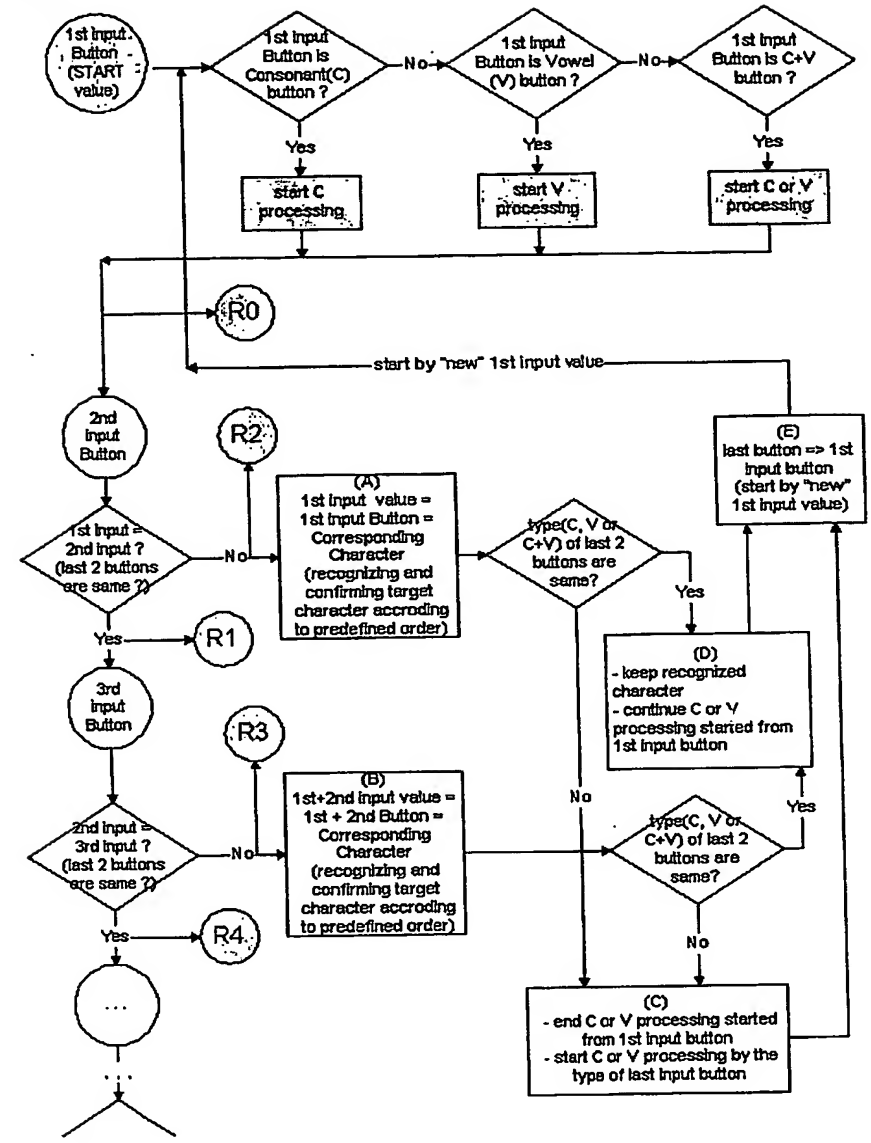

1st input Button (START value)
(1) 1st input Button is Consonant(C) button ?
No
(1) 1st input Button is Vowel(V) button ?
Yes
Yes
start C processing
start V processing
R0
start by "new" 1st input value
2nd input Button
R2
(A) 1st input value = 1st input Button = Corresponding Character (recognizing and confirming target character accroding to predefined order)
(2) 1st input = 2nd input ? (last 2 buttons are same ?)
No
Yes
R1
(4) type(C or V) of last 2 buttons are same?
Yes
(E) last button => 1st input button (start by "new" 1st input value)
(D) - keep recognized character - continue C or V processing started from 1st input button
3rd input Button
R3
(B) 1st+2nd input value = 1st + 2nd Button = Corresponding Character (recognizing and confirming target character accroding to predefined order)
No
(3) 2nd input = 3rd input ? (last 2 buttons are same ?)
No
(5) type(C or V) of last 2 buttons are same?
Yes
Yes
R4
No
(C) - end C or V processing started from 1st input button - start C or V processing by the type of last input button

【도 57】

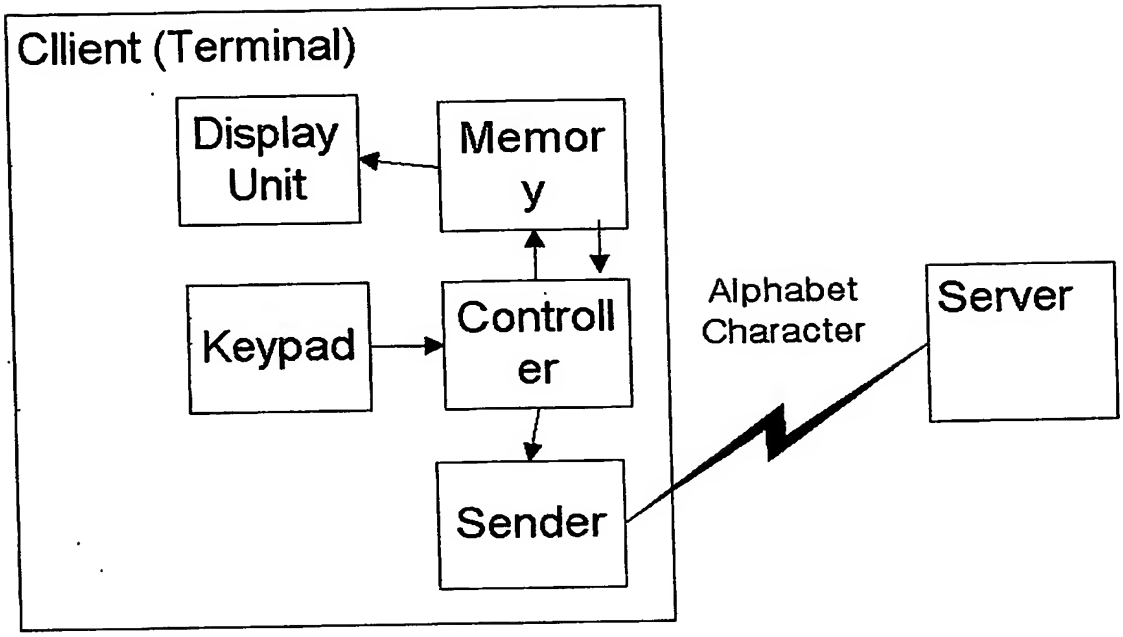
도 10-8


START
n <- 0
n <- (n+1)
(1)
n-th Input
Button
start by "new" 1st input value : n <- 1
n <- (n+1)
start C
processing
start
Vowel(V)
processing
Yes
n <- (n+1)
number of input
buttons (n) >=
2 ?
No
1st input Button
is
Consonant(C)
button ?
No
last button => 1st
input button
n <- 1
(start by "new"
1st input value)
Yes
Yes
(last 2 buttons
are same ?)
No
(A)
input buttons from
1st to n-1 th =
n-1 times presses of
same button =
Corresponding
Character
(recognizing and
confirming target
character accroding
to predefined order)
type(C or V) of
last 2 buttons
are same?
Yes
(D)
- keep recognized
character
- continue C or V
processing started
from 1st input button
No
(C)
- end C or V processing started
from 1st input button
- start C or V processing by the
type of last input button

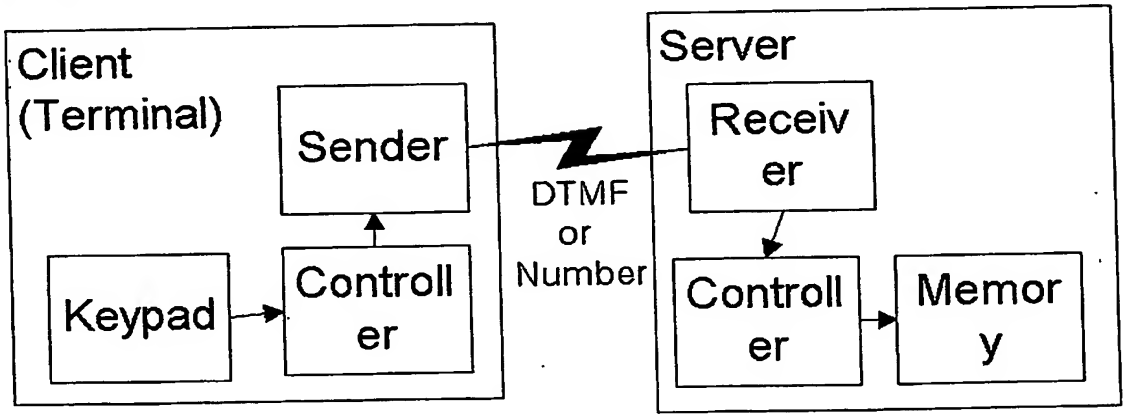
【도 58】
도 10-9


1st Input Button (START value)
1st Input Button is Consonant(C) button ?
No
1st Input Button is Vowel (V) button ?
No
1st Input Button is C+V button ?
Yes
start C processing
Yes
start V processing
Yes
start C or V processing
R0
start by "new" 1st input value
2nd Input Button
R2
(A)
1st input value = 1st input Button = Corresponding Character (recognizing and confirming target character accroding to predefined order)
type(C, V or C+V) of last 2 buttons are same?
Yes
(E)
last button => 1st input button (start by "new" 1st input value)
1st Input = 2nd Input ? (last 2 buttons are same ?)
No
Yes
R1
(D)
- keep recognized character
- continue C or V processing started from 1st input button
3rd Input Button
R3
(B)
1st+2nd input value = 1st + 2nd Button = Corresponding Character (recognizing and confirming target character accroding to predefined order)
No
type(C, V or C+V) of last 2 buttons are same?
Yes
2nd Input = 3rd Input ? (last 2 buttons are same ?)
No
Yes
R4
No
(C)
- end C or V processing started from 1st input button
- start C or V processing by the type of last input button

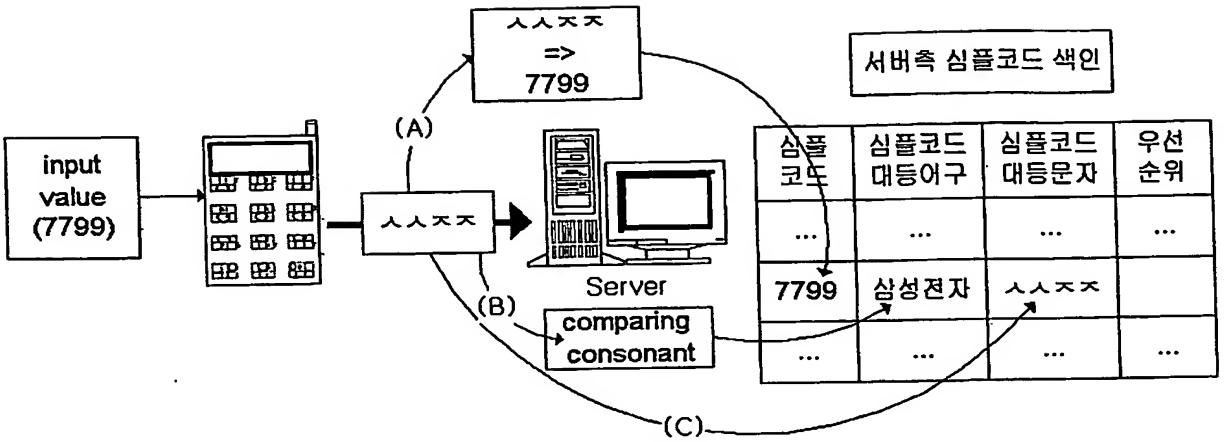
【도 45】
도 11-1



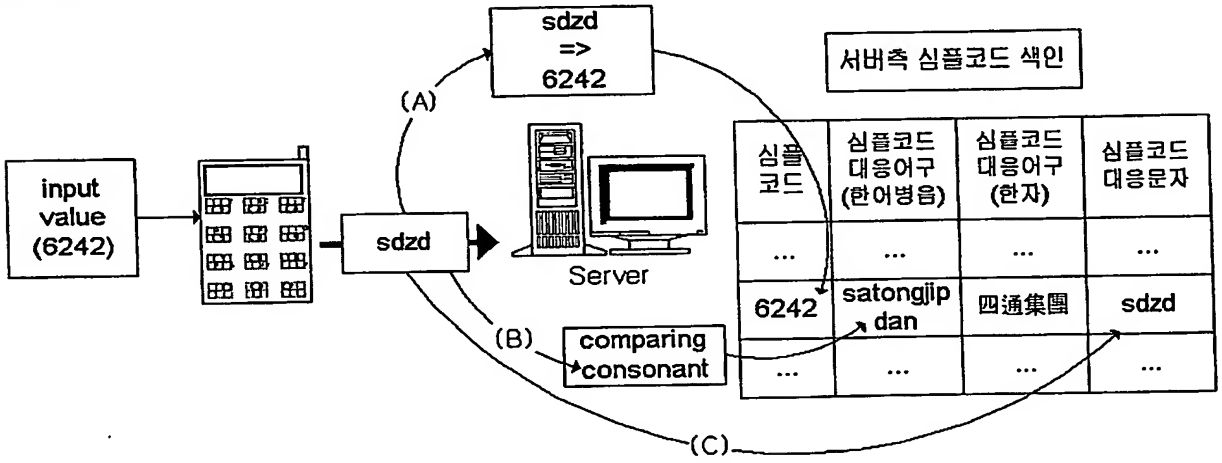
【도 46】
도 11-2



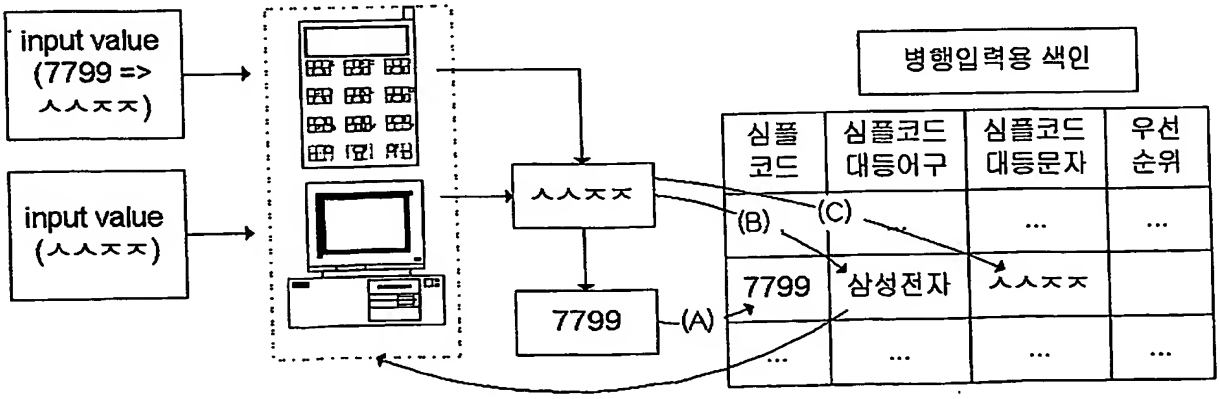
【도 47】
도 11-3



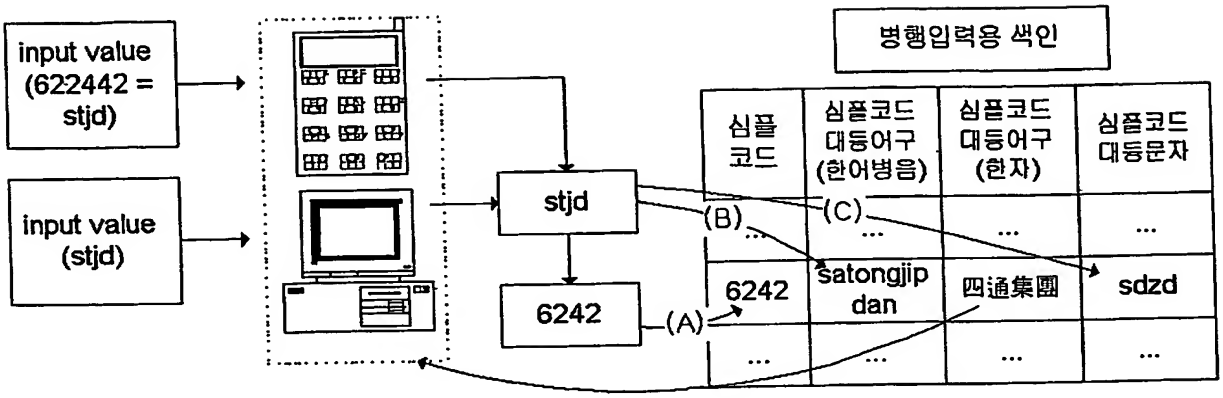
【도 48】
도 11-4



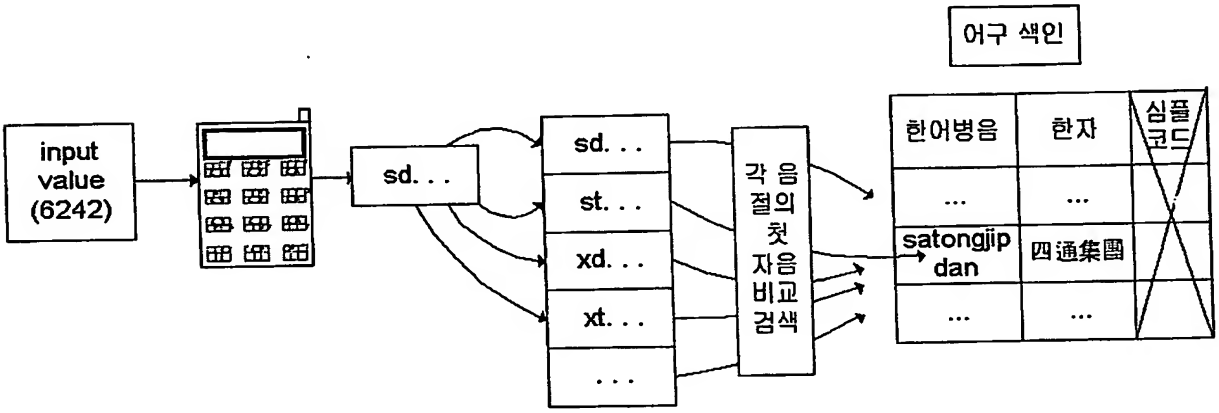
【도 49】
도 11-5



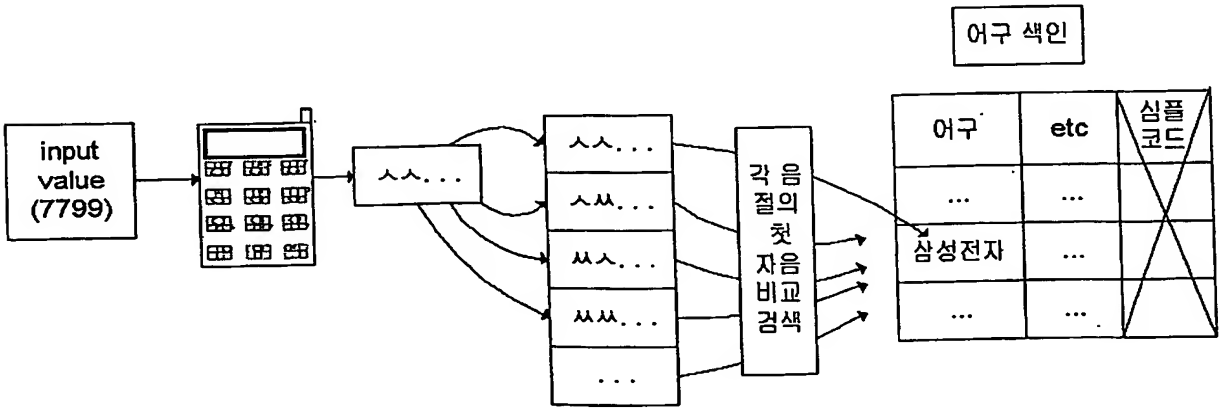
【도 50】
도 11-6



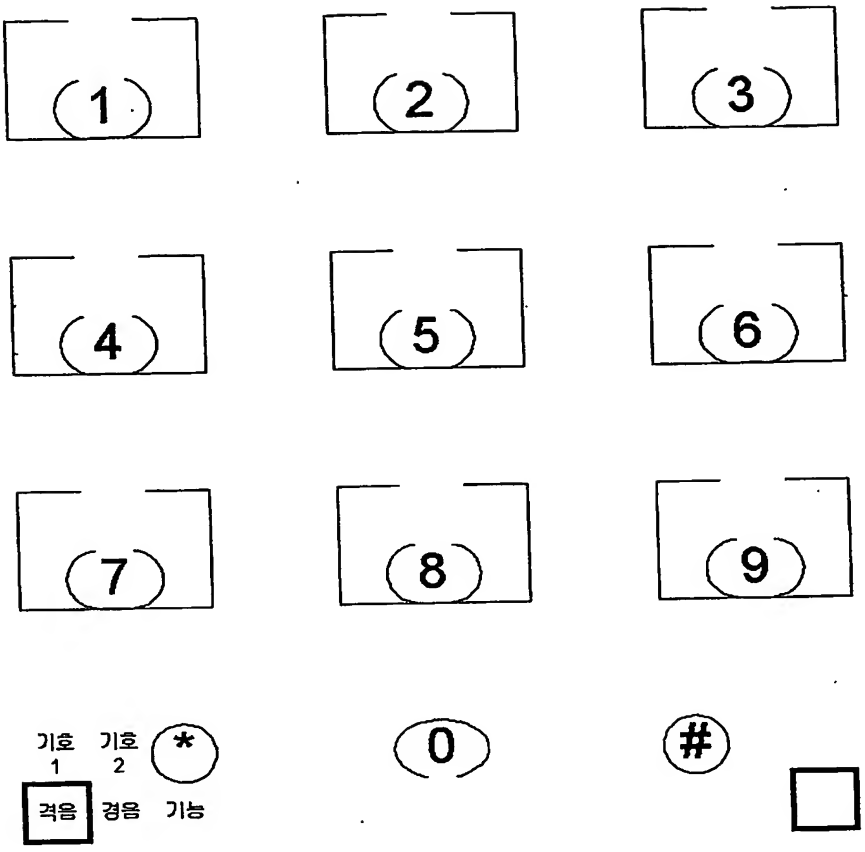
【도 51】
도 11-7



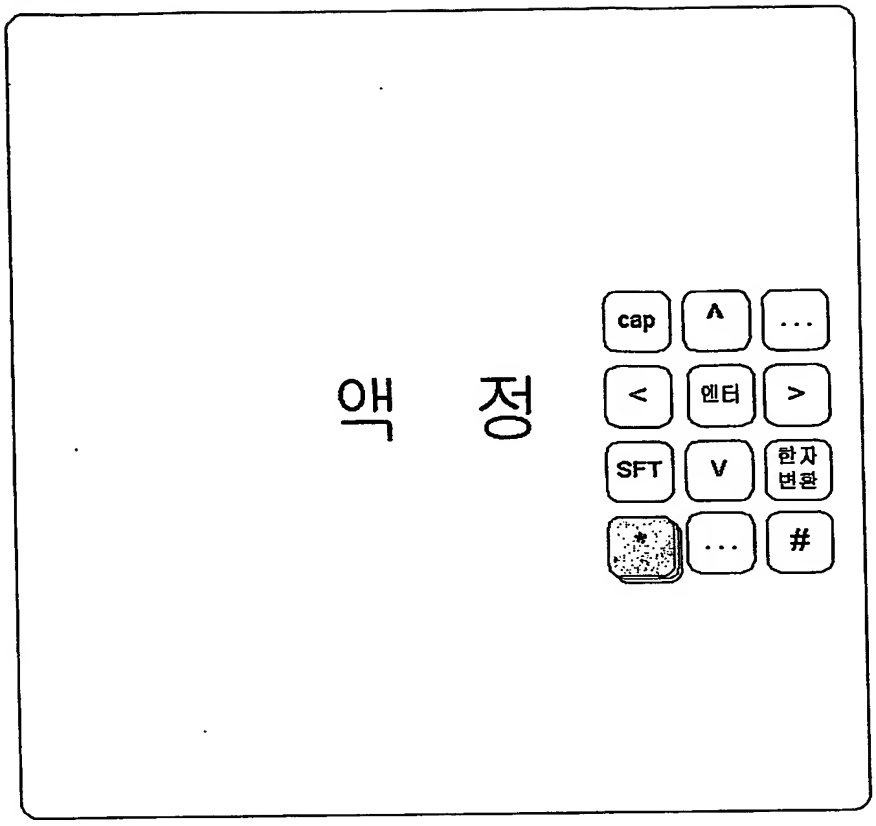
【도 52】
도 11-8



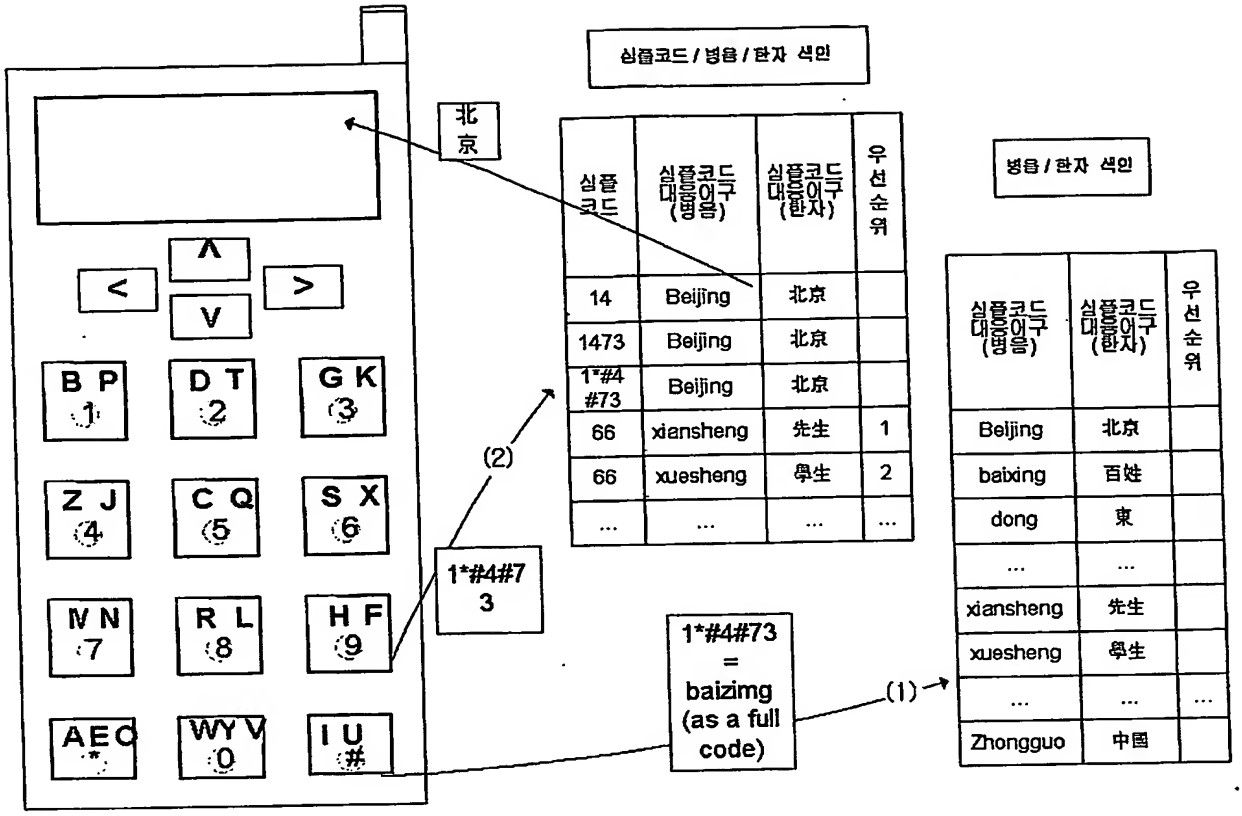
【도 53】
도 12-1



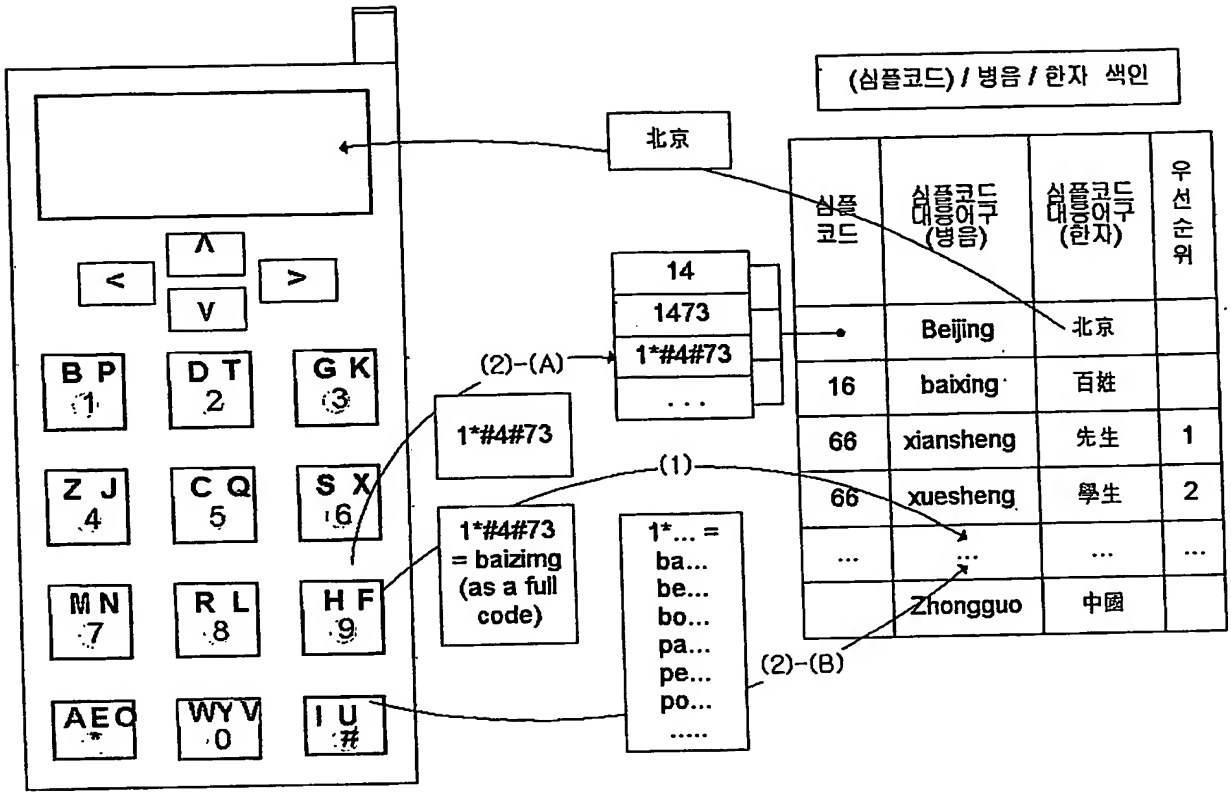
【도 54】
도 12-2



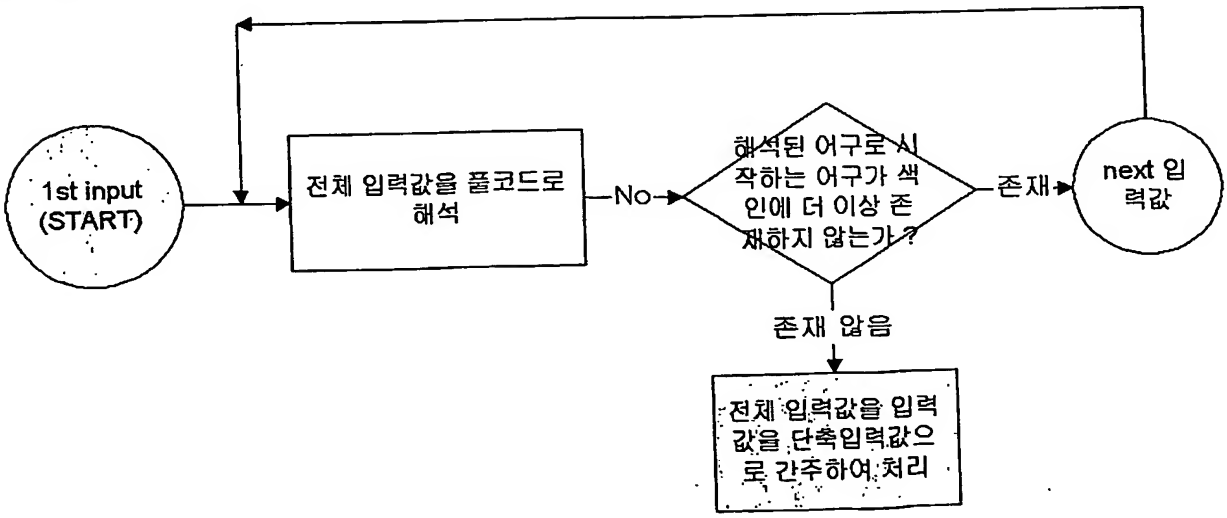
【도 59】
도 10-10


北京
심플코드 / 병음 / 한자 색인
심플코드
심플코드 대응어구 (병음)
심플코드 대응어구 (한자)
우선순위
14 Beijing 北京
1473 Beijing 北京
1*#4#73 Beijing 北京
66 xiansheng 先生 1
66 xuesheng 學生 2
병음 / 한자 색인
Beijing 北京
baixing 百姓
dong 東
xiansheng 先生
xuesheng 學生
Zhongguo 中国
1*#4#73
(2)
1*#4#73 = baizimg (as a full code)
(1)
A
V
<
>
B P 1
D T 2
G K 3
Z J 4
C Q 5
S X 6
M N 7
R L 8
H F 9
AEC *
WYV 0
I U #

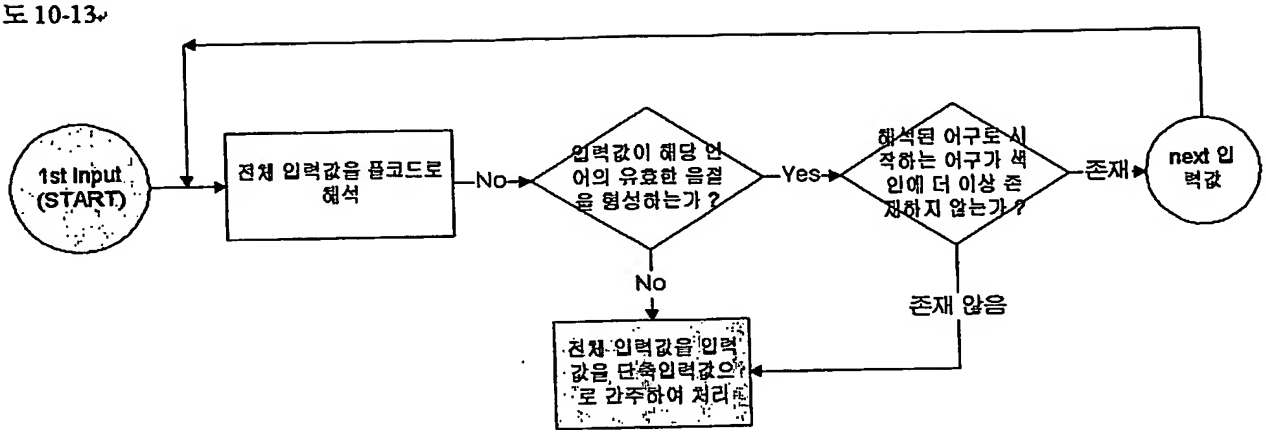
【도 60】
도 10-11



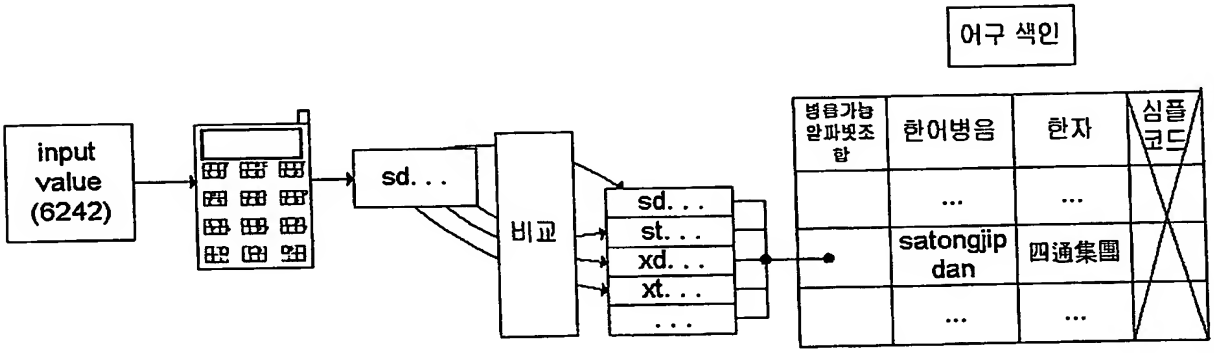
【도 61】
도 10-12



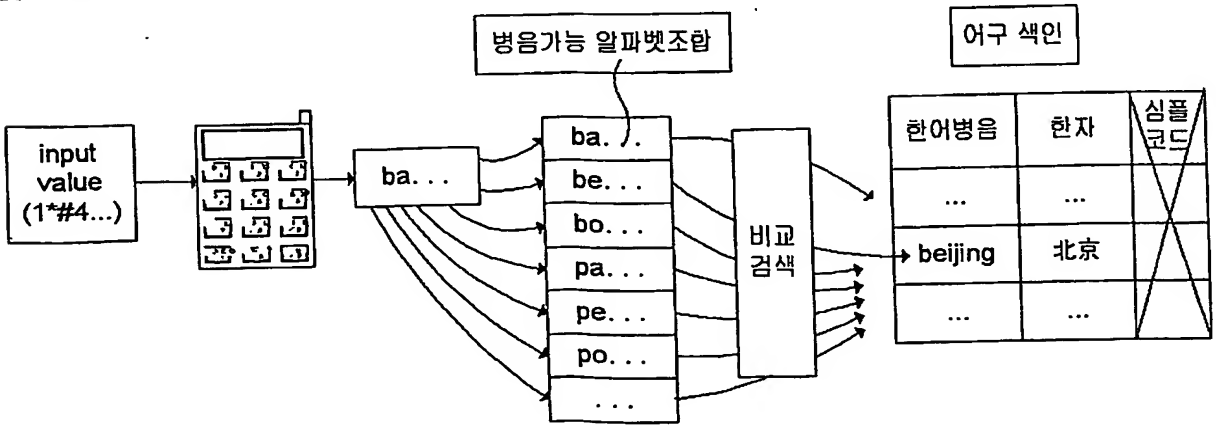
【도 62】
도 10-13



【도 63】
도 11-9



【도 64】
도 11-10



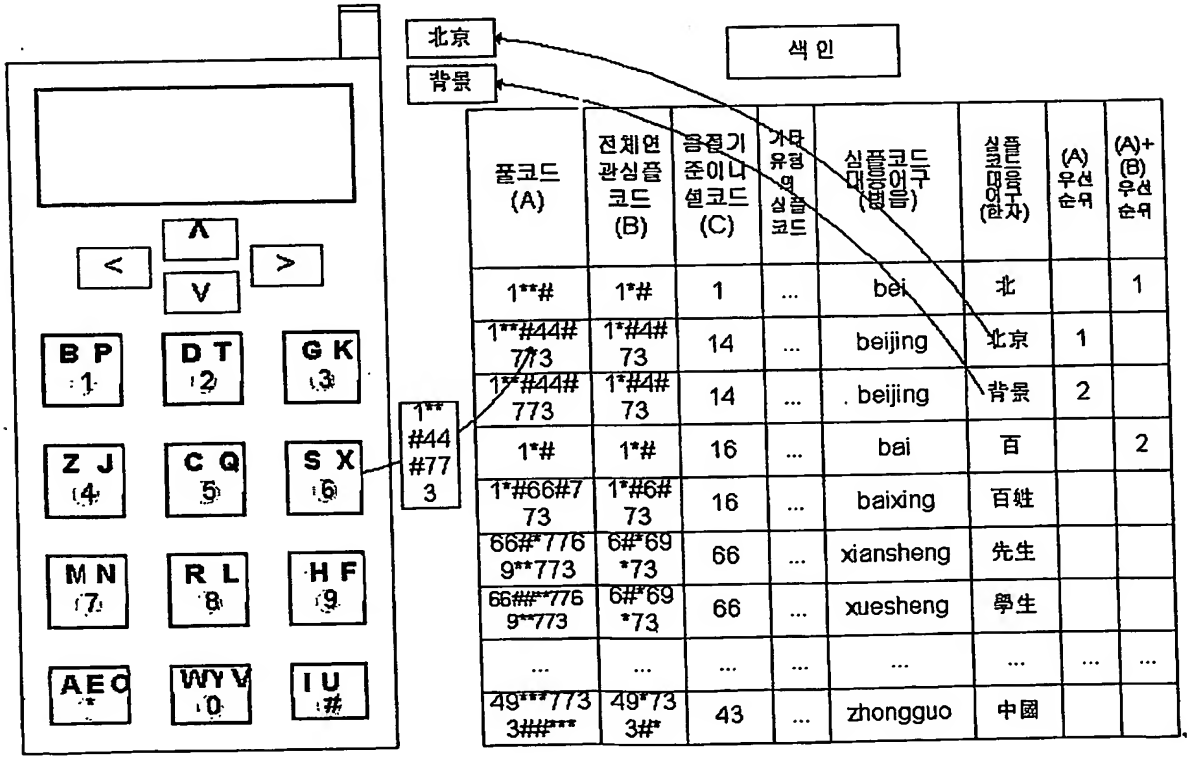
【도 65】

도 11-11

(A) 단축입력값 (즉, 심플코드 혹은 단축코드)			
(B) 부분연관 심플코드	(C) 음절기준이니셜코드	(H) 일부알파벳기준입력값 (심플코드)	(I) 음절기준이니셜값 (심플코드)
	(D) 첫모음+음절기준이니셜코드		(J) 첫모음+음절기준이니셜값 (심플코드)
	(E) 자음기준심플코드		(K) 자음기준입력값 (심플코드)
	(F) 단어기준이니셜코드		(L) 단어기준이니셜값 (심플코드)
(G) 전체연관심플코드		(Z)	

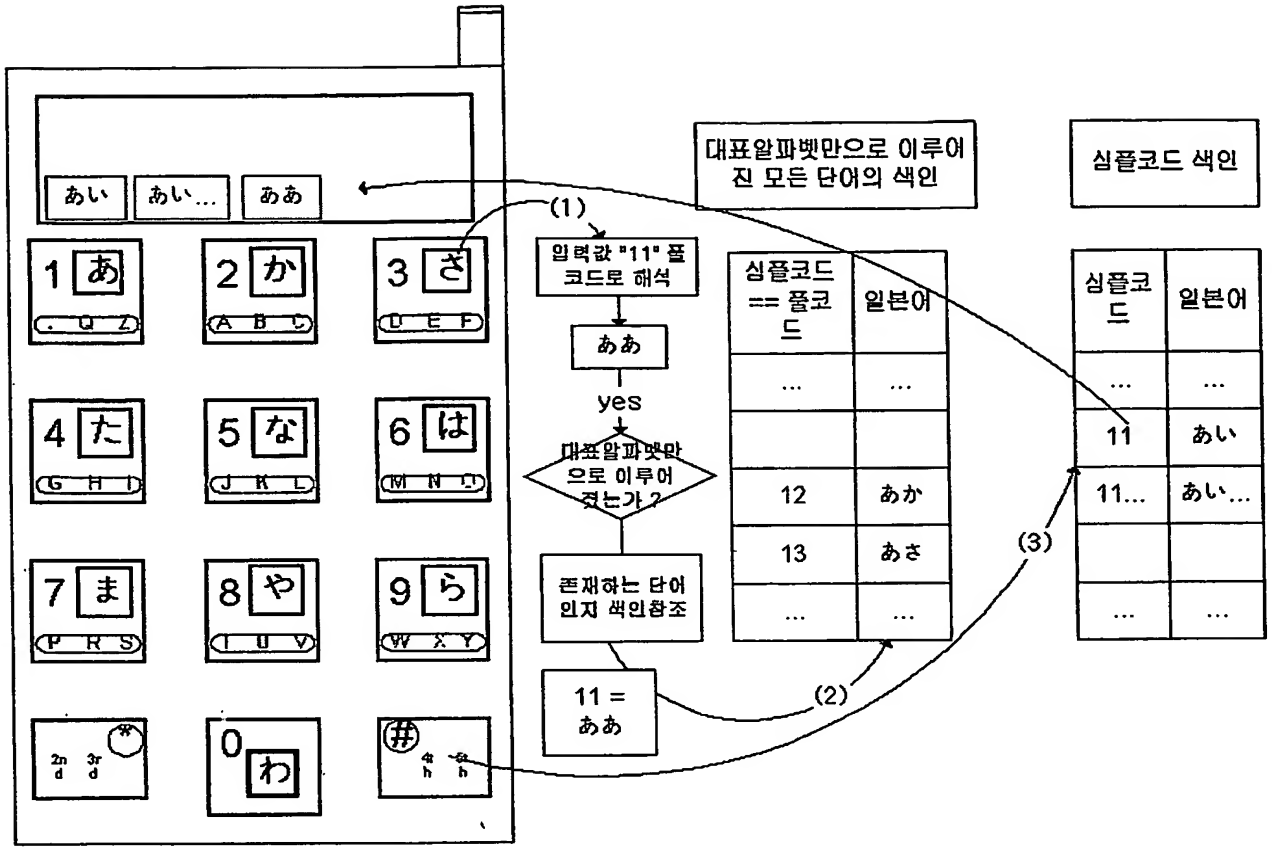
【도 66】

도 10-14

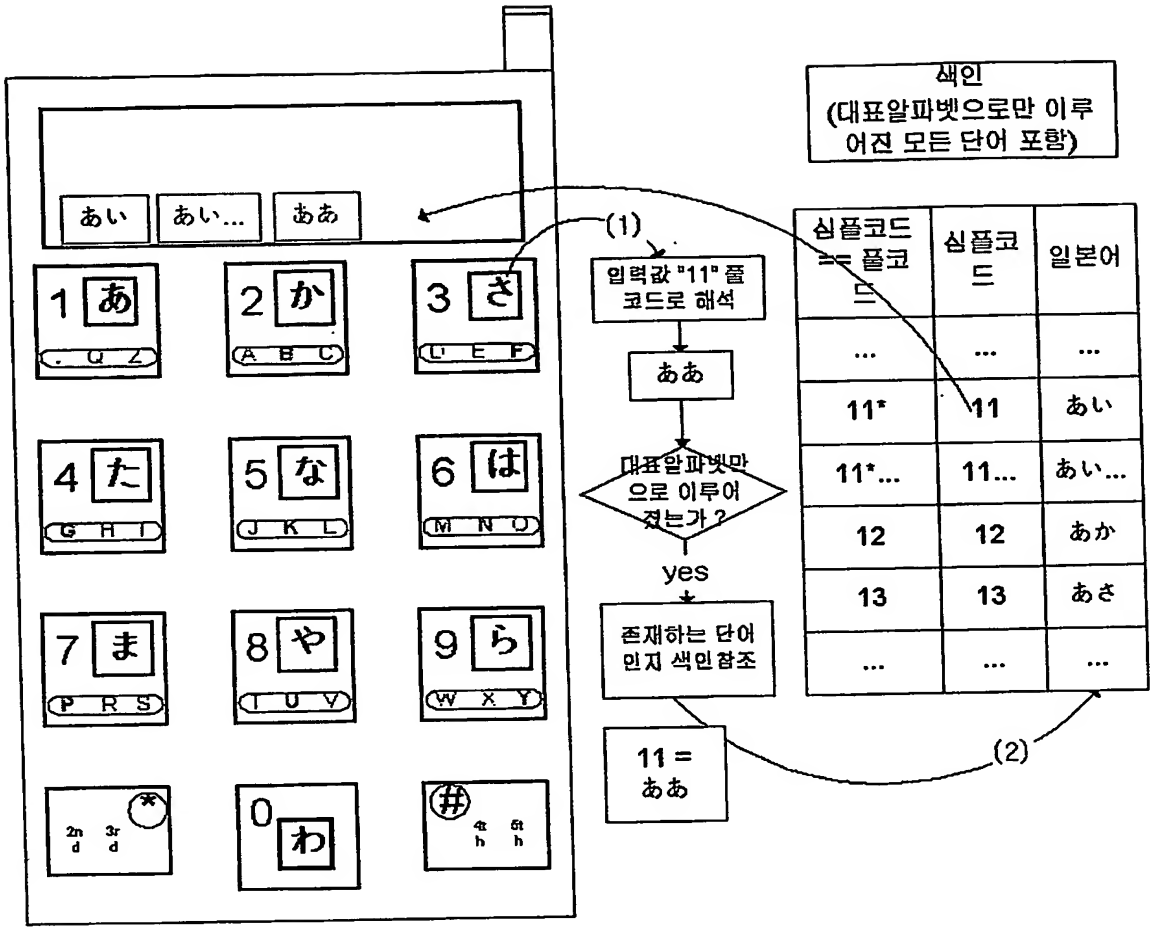


풀코드 (A)	전체연관심플코드 (B)	음절기준이니셜코드 (C)	가타유형의 심플코드	심플코드 대응어구 (병음)	심플코드 대응어구 (한자)	(A) 우선순위	(A)+(B) 우선순위
1**#	1*#	1	...	bei	北		1
1**#44#7773	1*#4#73	14	...	beijing	北京	1	
1**#44#773	1*#4#73	14	...	beijing	背景	2	
1*#	1*#	16	...	bai	百		2
1*#66#773	1*#6#73	16	...	baixing	百姓		
66#*7769**773	6#*69*73	66	...	xiansheng	先生		
66##*7769**773	6#*69*73	66	...	xuesheng	學生		
...	...	...	...	...	...	...	...
49***7733##***	49*733#*	43	...	zhongguo	中國		

【도 67】
도 10-15

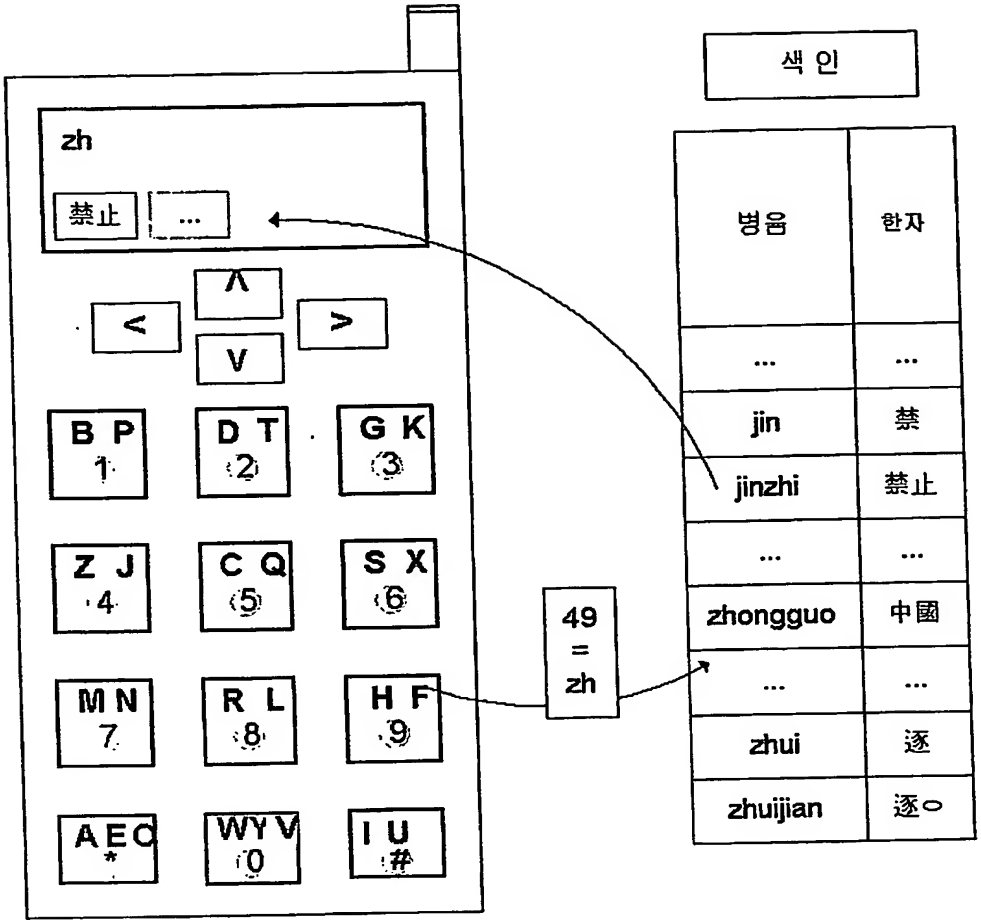

あい
あい...
ああ
1 あ
2 か
3 さ
4 た
5 な
6 は
7 ま
8 や
9 ら
0 わ
(1)
입력값 "11" 풀 코드로 해석
ああ
yes
대표알파벳만으로 이루어 졌는가?
존재하는 단어인지 색인참조
11 = ああ
(2)
(3)
대표알파벳만으로 이루어진 모든 단어의 색인
심플코드 == 풀코드
일본어
12
あか
13
あさ
심플코드 색인
심플코드
일본어
11
あい
11...
あい...

【도 68】
도 10-16

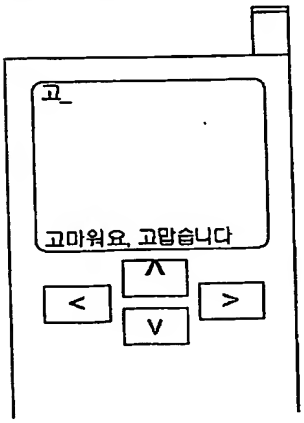

あい
あい...
ああ
1 あ . Q Z
2 か A B C
3 さ D E F
4 た G H I
5 な J K L
6 は M N O
7 ま P R S
8 や T U V
9 ら W X Y
0 わ
2n d 3r d
4t h 5t h
(1)
입력값 "11" 풀 코드로 해석
ああ
대표알파벳만으로 이루어 졌는가 ?
yes
존재하는 단어인지 색인참조
11 = ああ
(2)
색인
(대표알파벳으로만 이루어진 모든 단어 포함)
심플코드 == 풀코드
심플코드
일본어
... ...
11* 11 あい
11*... 11... あい...
12 12 あか
13 13 あさ
... ...

【도 69】

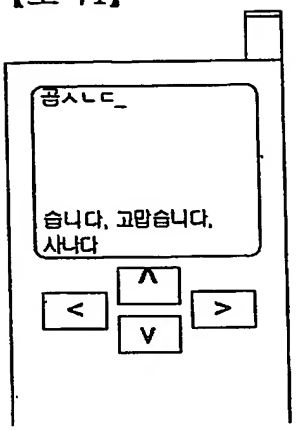
도 10-17



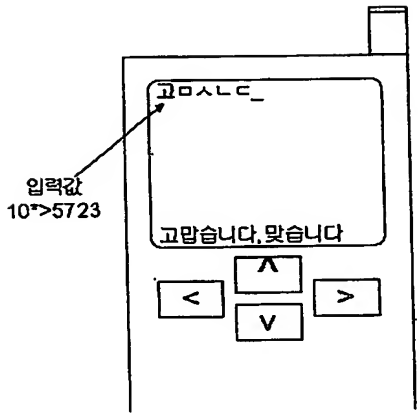
【도 70】



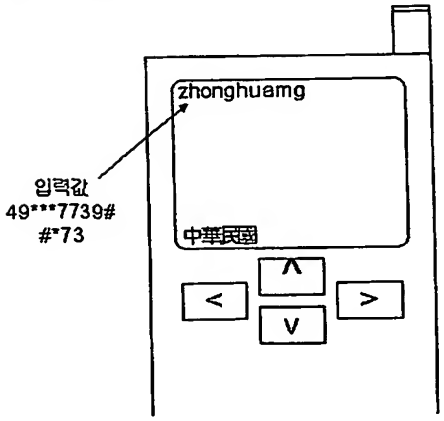
【도 71】


공ㅅㄴㄷ_
습니다. 고맙습니다.
사나다
^
<
>
v

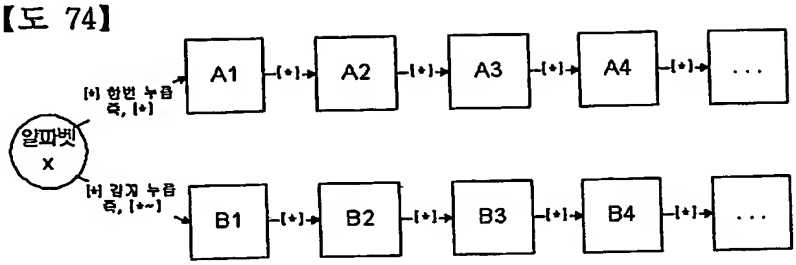
【도 72】


고ㅁㅅㄴㄷ_
입력값
10*>5723
고맙습니다.맞습니다
^
<
>
v

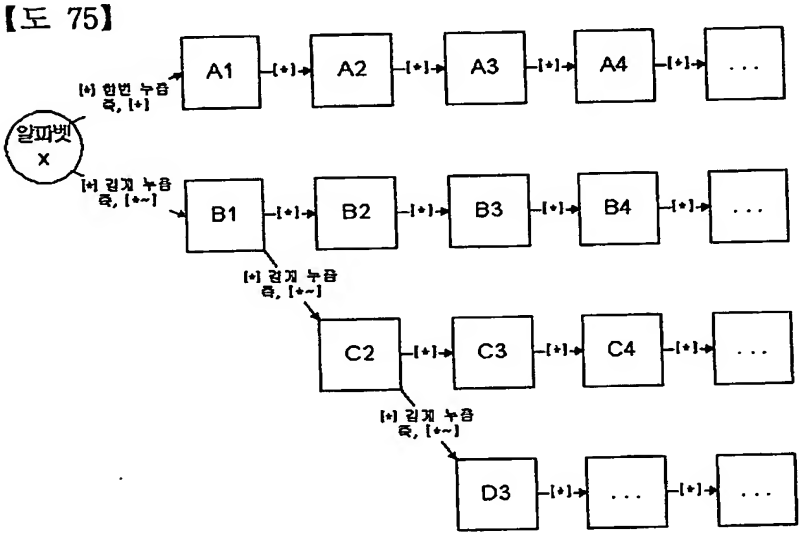
【도 73】


zhonghuamg
입력값
49***7739#
#*73
中華民國
^
<
>
v

【도 74】


A1
A2
A3
A4
...
[+] 한번 누름
즉, [+]
알파벳
X
[+] 길게 누름
즉, [+~]
B1
B2
B3
B4
...

【도 75】


A1
A2
A3
A4
...
[+] 한번 누름
즉, [+]
알파벳
X
[+] 길게 누름
즉, [+~]
B1
B2
B3
B4
...
[+] 길게 누름
즉, [+~]
C2
C3
C4
...
[+] 길게 누름
즉, [+~]
D3
...
...